Forschungsberichte aus der Industriellen Informationstechnik | 30

Johannes Anastasiadis

Überwachte Methoden für die spektrale Entmischung mit künstlichen neuronalen Netzen

KIT Scientific Publishing

Johannes Anastasiadis

Überwachte Methoden für die spektrale Entmischung mit künstlichen neuronalen Netzen

Forschungsberichte aus der Industriellen Informationstechnik
Band 30

Institut für Industrielle Informationstechnik
Karlsruher Institut für Technologie
Hrsg. Prof. Dr.-Ing. Michael Heizmann

Eine Übersicht aller bisher in dieser Schriftenreihe erschienenen Bände finden Sie am Ende des Buchs.

Überwachte Methoden für die spektrale Entmischung mit künstlichen neuronalen Netzen

von
Johannes Anastasiadis

Scientific Publishing

Karlsruher Institut für Technologie
Institut für Industrielle Informationstechnik

Überwachte Methoden für die spektrale Entmischung
mit künstlichen neuronalen Netzen

Zur Erlangung des akademischen Grades eines Doktor-Ingenieurs
von der KIT-Fakultät für Elektrotechnik und Informationstechnik des
Karlsruher Instituts für Technologie (KIT) genehmigte Dissertation

von Johannes Anastasiadis, M.Sc.

Tag der mündlichen Prüfung: 15. Mai 2023
Hauptreferent: Prof. Dr.-Ing. Michael Heizmann, KIT
Korreferent: Prof. Dr.-Ing. Uwe Sörgel, Universität Stuttgart

Impressum

KIT Scientific Publishing

Karlsruher Institut für Technologie (KIT)
KIT Scientific Publishing
Straße am Forum 2
D-76131 Karlsruhe

KIT Scientific Publishing is a registered trademark
of Karlsruhe Institute of Technology.
Reprint using the book cover is not allowed.

www.ksp.kit.edu

Print on Demand 2023 – Gedruckt auf FSC-zertifiziertem Papier

ISSN 2190-6629
ISBN 978-3-7315-1305-6
DOI 10.5445/KSP/1000159281

Vorwort

Diese Arbeit, die während meiner Zeit als wissenschaftlicher Mitarbeiter am Institut für Industrielle Informationstechnik (IIIT) des Karlsruher Instituts für Technologie (KIT) entstand, wurde in dieser Form durch die Unterstützung zahlreicher Personen ermöglicht.

Zuerst bedanke ich mich bei Prof. Dr.-Ing. Fernando Puente León, der bereits in meiner Zeit als Student mein Interesse für die Signalverarbeitung weckte und mich in der ersten Hälfte meiner Zeit am IIIT betreute. Ich werde ihn als kompetenten Wissenschaftler und engagierten Mentor in Erinnerung behalten.

Ich bedanke mich bei Prof. Dr.-Ing. Michael Heizmann für die hervorragende Betreuung während der restlichen Zeit. In diesen Jahren gab er mir wertvolle Impulse für meine Forschung und nahm sich stets Zeit für meine Anliegen und die meiner Kollegen. Außerdem möchte ich mich bei Prof. Dr.-Ing. Uwe Sörgel für sein Interesse an meiner Arbeit und die Übernahme des Korreferats bedanken.

Bei meinen Kollegen bedanke ich mich für die gute Zusammenarbeit und die gemeinsame Zeit am IIIT, die durch Hilfsbereitschaft und produktiven Austausch geprägt war. Ein besonderer Dank gilt Manuel Bihler, Daniel Diaz Ocampo, Muen Jin, Fabian Leven, Daniel Leyer, Lanxiao Li, Theresa Panther, Markus Schwabe, Johannes Steffens, Erik Tabuchi Barczak, David Uhlig und Hannes Weinreuter für das Korrekturlesen dieser Arbeit. Auch bedanke ich mich bei Manuela Moritz, Patricia Nestl, Dieter Brandt und Marvin Winkler für die Unterstützung in organisatorischen und technischen Belangen.

Zum Schluss möchte ich mich bei meinen Eltern Mariella und Anastasios und meinem Bruder Andreas für ihre bedingungslose Unterstützung und ganz besonders bei meiner Frau Katharina und meinem Sohn Aaron für ihre Geduld und die mentale Unterstützung bedanken.

Karlsruhe, Juni 2023 Johannes Anastasiadis

Inhaltsverzeichnis

Symbolverzeichnis

Allgemeine Abkürzungen

Abkürzung	**Bedeutung**
bzw.	beziehungsweise
bspw.	beispielsweise
d. h.	das heißt
engl.	englisch
vgl.	vergleiche
z. B.	zum Beispiel
AD	Albedo-Domäne
AMSA	*average minimal spectral angle*
AOTF	*acousto-optic tunable filter*
CGAN	*conditional Generative Adversarial Network*
CNN	*convolutional neural network*
ELMM	erweitertes lineares Mischmodell
FCLS	*Fully-Constrained-Least-Squares*-Verfahren
FM	Fan-Modell
GAN	*Generative Adversarial Network*
GBM	generalisiertes bilineares Mischmodell
GCNN	*generative convolutional neural network*
GCNN-2	...mit Verdopplung der zufälligen Eingänge bei der Datenerzeugung
GCNN-Cov	...mit Regularisierung durch die Kovarianzmatrix
GLMM	generalisiertes lineares Mischmodell
GPNN	Gauß-Prozess-inspirierte neuronale Netze
GPNN-M	...nur Mittelwertvektor verwendet
KNN	künstliches neuronales Netz
LMM	lineares Mischmodell

Abkürzung	Bedeutung
LQM	linear-quadratisches Mischmodell
MSE	*mean squared error*
MSSA	*mean spectra spectral angle*
NZ	normalverteilte Zufallsvektoren
ReLU	*rectified linear unit*
RGAN	regularisierendes *Generative Adversarial Network*
RGAN-DE	...mit Diskriminator bei der Datenerzeugung
RGB	Rot, Grün, Blau
RMSE	*root-mean-square error*
SA	*spectral angle*
ZR	zufällige Reinspektrenwahl

Symbole

Lateinische Buchstaben

Symbol	Bedeutung
$\mathbf{a}, \hat{\mathbf{a}}$	Anteilsvektor, Schätzung
$b, \mathbf{b}$	Schwellenwert (*bias*)
C	Anzahl Merkmalskarten bzw. Kanäle
$\mathcal{D}$	Diskriminator (GAN)
d	Partikeldurchmesser
e	Eulersche Zahl
$f, \mathbf{f}$	Funktion oder Modell
$\mathcal{G}$	Generator (GAN)
H	Chandrasekhars H-Funktion
I	Länge Faltungskern
J	Länge Eingangsvektor
k_{U}	Kovarianzfunktion
$\mathrm{k}_{\boldsymbol{\Sigma}}, \mathrm{k}_{\mathrm{R}}$	Gewichtungsfaktoren (Verlustfunktion)
K	Länge Ausgangsvektor
L, l	*Lagrange*-Funktion, -Multiplikator
ℓ	Verlustfunktion, Kostenfunktion
m_{U}	Kovarianzfunktion

Symbol	Bedeutung
$\mathbf{m}, \mathbf{M}, \hat{\mathbf{M}}$	Reinspektrum, Reinspektrenmatrix, Schätzung
$\mathcal{M}$	Reinspektrenmenge
$\mathbf{m}$	Reinspektrum als Zufallsvariable
N	Anzahl Datenpunkte
$\mathcal{N}$	neuronales Netz
$\mathcal{N}\{\bullet, \bullet\}$	Normalverteilung unter Angabe von Mittelwertvektor und Autokovarianzmatrix
P	Anzahl Reinstoffe
P_{A}	Leistung des absorbierten Lichts
P_{S}	Leistung des gestreuten Lichts
S	Anzahl Spektren in Spektrenmenge
s	Zufallsvariable für Index der Spektrenmenge
$\mathcal{S}$	Anteilsstufe
$\mathcal{U}\{\bullet, \bullet\}$	Gleichverteilung unter Angabe der Grenzen
v^{a}	Adam-Optimierer: gleitender quadratischer Mittelwert
$w, \mathbf{w}, \mathbf{W}$	Gewichte
$x_{\mathrm{x}}, x_{\mathrm{y}}$	Ortsindizes
$x, \mathbf{x}$	Eingangsdaten, Eingangsgrößen
$\mathcal{X}$	Datensatz
$\mathcal{X}_{\mathrm{Q3}}, \mathcal{X}_{\mathrm{Q4}}, \mathcal{X}_{\mathrm{F4}}$	Datensätze: Quarzsande mit 3 bzw. 4 Komponenten, Farbpulver mit 4 Komponenten
$y, \mathbf{y}$	Ausgangsdaten, Ausgangsgrößen
Z	Länge Zufallsvektor
$\mathrm{z}, \mathbf{z}$	Zufallsvariable, Zufallsvektor

Griechische Buchstaben

Symbol	Bedeutung
α	Lernrate
β, β_0	Winkel zwischen gestreutem bzw. einfallendem Licht und Normalenvektor
β_1, β_2	Parameter Adam-Optimierer
γ	Nichtlinearitätskoeffizient GBM
Γ_{bi}	bidirektionale Reflektanz

Symbol	Bedeutung
Γ_{rel}	relative Reflektanz
δ	Nichtlinearitätskoeffizient LQM
ϵ	Fehlermaß
ζ_1, ζ_2	Parameter RGAN-DE
$\theta, \boldsymbol{\theta}, \hat{\boldsymbol{\theta}}$	Parameter eines Optimierungsproblems, Schätzung
$\vartheta, \boldsymbol{\vartheta}$	dazugehöriger Gradient
ι, ι_0	Cosinus des Winkels zwischen gestreutem bzw. einfallendem Licht und Normalenvektor
κ	Porositätskoeffizient
$\tilde{\lambda}_1, \tilde{\lambda}_2$	transformierte Wellenlängenindizes
Λ	Anzahl Wellenlängenkanäle
μ^{a}	Adam-Optimierer: gleitender Mittelwert
$\boldsymbol{\mu}, \hat{\boldsymbol{\mu}}$	Mittelwertvektor, Schätzung
$\hat{\mu}_{\mathcal{B}}$	Stichprobenmittelwert *Batch*-Normalisierung
π	Kreiszahl
ρ	Dichte
$\hat{\sigma}_{\mathcal{B}}$	Stichprobenvarianz *Batch*-Normalisierung
$\boldsymbol{\sigma}$	Varianzvektor (VAE)
$\boldsymbol{\Sigma}, \hat{\boldsymbol{\Sigma}}$	Kovarianzmatrix, Schätzung
$\boldsymbol{\upsilon}, \hat{\boldsymbol{\upsilon}}$	Spektrum, erzeugtes Spektrum (Schätzung)
$Y, \hat{Y}$	Menge an Spektren eines Stoffgemischs, Schätzung
$\mho$	Spektrum als Gauß-Prozess
$\boldsymbol{\mho}$	Spektrum als Zufallsvektor
ϕ	Phasenwinkel
$\boldsymbol{\phi}$	Parametervektor ELMM
Φ	Phasenfunktion
$\boldsymbol{\Phi}$	Parametermatrix GLMM
$\varphi, \boldsymbol{\varphi}$	Aktivierungsfunktion
Ψ	Massenanteil
$\omega, \boldsymbol{\omega}$	Einzelstreuungsalbedo über Volumen gemittelt

(hochgestellte) Indizes

Index	Bedeutung
$(\bullet)^{\dagger}$	Pseudoinverse
$(\bullet)^{\mathrm{T}}$	Transposition
$(\bullet)^{*}$	wahrer Wert

Indizes

Index	Bedeutung
$(\bullet)_{\mathrm{a}}$	Ausgangs…
$(\bullet)_{\mathrm{e}}$	Eingangs…
$(\bullet)_{i}$	Index Faltungskern
$(\bullet)_{j}$	Index Eingangsvektor
$(\bullet)_{k}$	Index Ausgangsvektor
$(\bullet)_{n}$	Index eines Datenpunkts
$(\bullet)_{p}$	Reinstoffindex
$(\bullet)_{q}$	alternativer Reinstoffindex
$(\bullet)_{r}$	weiterer alternativer Reinstoffindex
$(\bullet)_{s}$	Index Spektrenmenge
$(\bullet)_{t}$	Zeitindex, Trainingsschritt
$(\bullet)_{z}$	Index eines Zufallsvektors
$(\bullet)_{\lambda}$	Wellenlängenindex

Mathematische Operatoren

Operator	Bedeutung
Cov	Kovarianz
$\mathcal{d}$	Vektor $\leftrightarrow$ Diagonalmatrix
E	Erwartungswert
$\odot$	elementweise Multiplikation

Operator	**Bedeutung**
$*$	Faltung
$\|\bullet\|_F^2$	Frobeniusnorm
$\langle \bullet, \bullet \rangle$	Innenprodukt
$\boldsymbol{\nabla}$	Nabla-Operator
$\circ$	Verkettung

1 Einleitung

Die Nutzung von Hyperspektralbildern wurde ursprünglich vor allem in der Fernerkundung [34] angewendet, bei der die Erde mit hyperspektralen Kameras beobachtet wird, die an Satelliten, Flugzeugen oder auch an Drohnen befestigt sind [6, 106]. Dabei gibt es verschiedene Anwendungen, die Nutzen aus den Informationen in den hyperspektralen Daten ziehen. Beispiele sind die Beobachtung von Vegetation [18, 129] sowie Unterstützung bei Landwirtschaft [57, 88] und Bergbau [2, 78].

Auch in der Industrie und in Laboren werden Hyperspektralbilder dank gesunkener Hardwarekosten und Weiterentwicklungen in der Kameratechnik genutzt. Beispiele dafür sind medizinische Anwendungen [74], die Sicherstellung der Qualität von Lebensmitteln [29, 37, 107, 117] sowie die Materialsortierung [128]. Sie kommen zum Einsatz, wenn Materialeigenschaften untersucht werden, für die in Farbbildern nicht genug Information enthalten ist. Als optisches Verfahren haben sie den Vorteil, dass sie berührungslos und nicht-destruktiv sind.

Aufgrund der Distanz zum beobachteten Objekt oder sehr kleiner Objektstrukturen sind häufig mehrere Substanzen im Bereich eines Pixels vorhanden. Besteht ein Interesse an deren Anteilen, kann die spektrale Entmischung eingesetzt werden. Auch die industrielle Nutzung von spektraler Entmischung ist denkbar, wenn es sich beim Produkt um kleine Einheiten handelt, sodass Stoffgemische im Bereich eines Pixels vorkommen. Dies wäre bspw. bei Pulvermischungen wie Backmischungen, Fertiggerichten, Medikamenten und feinem Schüttgut der Fall.

1.1 Problemstellung

In dieser Arbeit steht die spektrale Entmischung von Stoffgemischen im Mittelpunkt. Ziel der spektralen Entmischung ist es, die relativen Anteile der Reinstoffe aus dem Spektrum eines Stoffgemischs (Mischspektrum)

zu erhalten [60]. Oft werden bei der spektralen Entmischung auch die Spektren der Reinstoffe (Reinspektren) ermittelt. Die Reinspektren werden in dieser Arbeit als bekannt vorausgesetzt, da diese ohnehin als Trainingsdaten für die vorgestellten überwachten Verfahren zur spektralen Entmischung benötigt werden. Um die Anteile der beteiligten Stoffe aus einem gemessenen Spektrum zu erhalten, nutzen klassische Ansätze ein Mischmodell. Dieses beschreibt, wie sich ein Mischspektrum aus den beteiligten Reinspektren in Abhängigkeit der relativen Anteile zusammensetzt [23]. Die Ermittlung der Anteile erfolgt durch Minimierung des Fehlers zwischen Modell und gemessenem Spektrum [22, 47, 49].

Bei den gängigen Mischmodellen, wie bspw. dem linearen Mischmodell, handelt es sich um vereinfachte Beschreibungen der Zusammenhänge [60]. Tatsächlich haben sehr viele Größen Einfluss auf das gemessene Spektrum [10, 43]. Beispiele sind die Winkel zwischen Oberfläche, Beleuchtung und Kamera, die geometrischen Eigenschaften der Partikel und Schatten. Diese führen dazu, dass Reinspektren und Mischspektren mit gleichen Anteilen der Reinstoffe unterschiedlich ausfallen können. Die meisten dieser Größen sind in der Praxis oft unbekannt, sodass deren Auswirkungen als Spektrenvariabilität zusammengefasst werden. Diese Variabilität erschwert die spektrale Entmischung und kann bei der Modellierung durch zusätzliche Parameter, wie Skalierungsfaktoren für die Reinspektren, berücksichtigt werden [54, 132].

Künstliche neuronale Netze (KNN) konnten in der jüngsten Vergangenheit große Erfolge erzielen [35], bspw. bei der Klassifikation und Segmentierung von Bildern oder der Spracherkennung [85, 105]. Einige Ansätze bieten auch die Möglichkeit, KNN zur Synthese von Daten zu nutzen [36, 51, 62]. Beim Einsatz für die spektrale Entmischung haben KNN den Vorteil, dass die Zusammenhänge zwischen gemessenen Spektren und den Anteilen datenbasiert gelernt werden können, ohne dass eine Modellierung notwendig ist. Dies gelingt nur, wenn ausreichend Trainingsdaten zur Verfügung stehen, was im industriellen Umfeld oft nicht der Fall ist, sodass sie aufwendig erstellt werden müssen.

Die meisten KNN, die aktuell zur spektralen Entmischung eingesetzt werden, werden unüberwacht trainiert. Das heißt, es werden nur die Spektren und nicht die dazugehörigen Anteile zum Training benötigt (siehe Kapitel 4). Dazu wird ein *Autoencoder* verwendet, der im ersten Teil

die Spektren auf wenige Werte (Anzahl der Reinstoffe) zusammenfasst und daraus im zweiten Teil die Spektren rekonstruiert. Damit an der Engstelle tatsächlich die Anteile herauskommen, wird der zweite Teil des Netzes so ausgelegt, dass er einem Mischmodell entspricht, was jedoch eine Einschränkung darstellt.

1.2 Eigener Beitrag

In dieser Arbeit werden Verfahren vorgestellt, bei denen der Zusammenhang zwischen gemessenem Spektrum und den relativen Anteilen der Reinstoffe überwacht aus Messdaten gelernt wird. Dazu werden Trainingsdaten benötigt, bei denen zu jedem Spektrum ein Anteilsvektor existiert, der die Anteile aller Reinstoffe angibt. Die zentralen Herausforderungen bei der spektralen Entmischung sind hierbei die bereits erwähnte Spektrenvariabilität und die Anteile als kontinuierliche Ausgangsgrößen. Beides sollte durch die Trainingsdaten abgedeckt sein.

Da es sehr aufwendig ist, reale Trainingsdaten zu erzeugen, werden Verfahren vorgestellt und untersucht, die basierend auf wenigen realen Trainingsdaten weitere erzeugen können. Dabei wird der Schwerpunkt auf eine Nutzung der spektralen Entmischung im industriellen Umfeld gelegt. Dies hat Auswirkungen auf die Beschaffenheit der Daten. Zum einen sind die beteiligten Reinstoffe bekannt, zum anderen wird davon ausgegangen, dass ein Trainingsdatensatz mit unterschiedlichen Stoffgemischen mit vertretbarem Aufwand erstellt werden kann. Dabei ist die Erstellung von Mischungen aufwendiger als die Anfertigung unterschiedlicher Aufnahmen derselben Mischung. So besitzen die untersuchten Datensätze viele Einzelspektren zu jedem Anteilsvektor. Damit lassen sich für die gegebenen Aufnahmebedingungen die statistischen Eigenschaften der Spektrenvariabilität ermitteln, die in den Verfahren zur Datenerzeugung und Datenerweiterung genutzt werden. Die vorgestellten Verfahren betrachten zwei unterschiedliche Ausgangssituationen. Im ersten Fall stehen nur Reinspektren zur Verfügung. Im zweiten Fall existiert ein Trainingsdatensatz, der auch Mischspektren beinhaltet.

Stehen nur Mengen an realen Reinspektren zur Verfügung, werden zur Datenerzeugung weiterhin Mischmodelle benötigt. Hier werden zwei Verfahren vorgestellt, die mit Hilfe der Reinspektren für vorgegebene

Anteile modellbasiert Mischspektren erzeugen. Dabei wird die Spektrenvariabilität der Mischspektren anhand der Spektrenvariabilität der vorhandenen Reinspektren abgeleitet. Die erzeugten Mischspektren werden anschließend genutzt, um ein KNN für die spektrale Entmischung zu trainieren. Neben der Möglichkeit der Berücksichtigung der Spektrenvariabilität hat dieser Ansatz den Vorteil, dass die Nebenbedingungen (Anteile immer positiv und in Summe 1) durch die Architektur des KNNs erzwungen werden können. Die beiden Verfahren unterscheiden sich dahingehend, wie die Spektrenvariabilität der Mischspektren zustande kommt. Der erste Ansatz nutzt bei der Mischung der Spektren je ein zufällig ausgewähltes Spektrum jedes beteiligten Reinstoffs. Beim zweiten Verfahren werden die Reinspektren als normalverteilte Zufallsvektoren modelliert. Dabei wird eine stochastische Formulierung von Mischmodellen verwendet, um die Zufallsvektoren der Mischspektren zu ermitteln. Einzelne Spektren sind dann zufällige Realisierungen dieser Zufallsvektoren. Allerdings bleibt bei beiden Verfahren die Einschränkung durch ein festzulegendes Modell bestehen.

Sind im ursprünglichen Trainingsdatensatz auch Spektren von Stoffgemischen enthalten, kann damit ein KNN rein datenbasiert trainiert werden. Um trotz weniger echter Stoffgemische seine Leistung zu verbessern, besteht die Möglichkeit, den vorhandenen Datensatz zu erweitern bzw. zu augmentieren. Typischerweise wird Datenaugmentierung genutzt, um damit Datenpunkte für bekannte Klassen zu erzeugen, die sich in bestimmten Eigenschaften unterscheiden, wie z. B. der Rotation bei Bildern. Damit kann eine Invarianz gegenüber dieser Eigenschaften erreicht werden, was vor allem bei der Klassifikation in der Bildverarbeitung sehr nützlich ist.

Bei der spektralen Entmischung wäre es wünschenswert, wenn das verwendete KNN eine Invarianz gegenüber der Spektrenvariabilität aufweisen würde. Darüber hinaus soll das KNN sinnvolle Ergebnisse für beliebige plausible Mischverhältnisse liefern. Diese können nicht alle als Trainingsdaten vorliegen, da es sich um eine kontinuierliche Ausgangsgröße handelt. Um diesem Problem entgegenzuwirken, werden die Datensätze dahingehend erweitert, dass Daten basierend auf Anteilsvektoren erzeugt werden, die nicht im ursprünglichen Trainingsdatensatz enthalten sind, wobei die Spektrenvariabilität berücksichtigt wird. Damit

sollen die kontinuierlichen Werte der Anteile besser abgedeckt werden. Zur Realisierung werden drei Verfahren entwickelt.

Eines basiert auf einem Faltungsnetz, das die Anteilsvektoren als Eingangsgrößen und die Spektren als Ausgangsgrößen hat. Hier wird die Spektrenvariabilität ausgenutzt, wodurch für jeden Eingangswert unterschiedliche Ausgangswerte zur Verfügung stehen. Damit für gleiche Anteilsvektoren unterschiedliche Spektren synthetisiert werden können, werden zusätzlich zufällige Eingangsgrößen genutzt. Die dadurch gewonnene Varianz in den Daten wirkt sich positiv auf das Training des generativen Faltungsnetzes aus, bei dem eine Überanpassung kaum ein Problem darstellt. Ein zweiter Ansatz integriert dieses Netz in ein *Generative Adversarial Network* (GAN), eine Klasse von künstlichen neuronalen Netzen, die zur Synthese von Daten genutzt werden kann. Ein dritter Ansatz modelliert die Spektren als Gauß-Prozesse und nutzt kleinere KNN, um den Zusammenhang zwischen den Anteilsvektoren, der Wellenlänge und den Momenten der Gauß-Prozesse abzubilden. Alle drei Verfahren werden mit den vorhandenen Trainingsdaten trainiert. Im Anschluss werden für zusätzliche Anteilsvektoren weitere Trainingsdaten erzeugt. Beim Verfahren, das die Spektren als Gauß-Prozesse modelliert, werden die Momente der Gauß-Prozesse der Mischspektren erzeugt. Die Spektren sind dann zufällig erzeugte Musterfunktionen jener Gauß-Prozesse.

Die Auswertung erfolgt mit Hilfe eines Faltungsnetzes und einem Vergleich mit etablierten Verfahren der spektralen Entmischung. Das Faltungsnetz ist ebenfalls im Laufe dieser Arbeit entstanden und wird mit Daten trainiert, die mit den vorgestellten Verfahren erzeugt werden. So kann der Unterschied der Leistung bei der spektralen Entmischung ermittelt werden, wobei ein Training mit dem ursprünglichen Trainingsdatensatz als Referenz dient.

1.3 Gliederung der Arbeit

Diese Arbeit ist folgendermaßen gegliedert: In Kapitel 2 werden die Grundlagen zur spektralen Entmischung dargestellt. Zunächst wird das Konzept des Hyperspektralbilds erläutert, was die Datengrundlage dieser Arbeit ist. Als nächstes wird der Begriff der Reflektanz definiert. Bei den hier gemessenen und verarbeiteten Spektren handelt es sich

um Reflektanzspektren. Danach werden die verwendeten Mischmodelle vorgestellt. Im Anschluss wird auf die Spektrenvariabilität eingegangen, bevor zum Schluss modellbasierte Verfahren zur spektralen Entmischung vorgestellt werden.

In Kapitel 3 werden die Grundlagen zu KNN, dem Werkzeug zur Verarbeitung der Daten in dieser Arbeit, vorgestellt. Dabei wird zunächst auf maschinelles Lernen im Allgemeinen eingegangen, bevor die KNN genauer beschrieben werden. Es folgen Abschnitte zu spezielleren Netzarchitekturen, die in dieser Arbeit von Bedeutung sind.

Im Anschluss werden in Kapitel 4 bereits existierende Verfahren vorgestellt, welche die spektrale Entmischung mit KNN durchführen, wobei die Unterschiede zu den hier vorgestellten Verfahren herausgestellt werden. Danach wird in Kapitel 5 die Netzarchitektur dargestellt, die hier zur spektralen Entmischung verwendet wird. Diese wird auch zum Vergleich der unterschiedlichen Datenerzeugungs- und Datenerweiterungsverfahren herangezogen. In den Kapiteln 6 und 7 werden jene Verfahren im Detail beschrieben.

Eine ausführliche Auswertung der Verfahren findet sich in Kapitel 8. Dort werden erst die verwendeten Datensätze vorgestellt, bevor auf die Ergebnisse eingegangen wird. Dabei werden die Ergebnisse in zwei Teile gegliedert: Zuerst werden nur die durch die Augmentierungsverfahren erzeugten Spektren an sich bewertet. Im Anschluss werden alle Verfahren bezüglich ihrer Leistung bei der spektralen Entmischung bewertet und verglichen, bevor alle Ergebnisse noch einmal zusammengefasst werden.

Die gesamte Arbeit wird schließlich in Kapitel 9 zusammengefasst und es wird ein Ausblick auf mögliche zukünftige Forschungsthemen gegeben.

2 Spektrale Entmischung

Die spektrale Entmischung ist die zentrale Anwendung dieser Arbeit. In diesem Kapitel werden die Grundlagen dazu vorgestellt und wichtige Begriffe eingeführt. Dazu werden in Abschnitt 2.1 Darstellung und Aufnahme von Hyperspektralbildern, die mit Hilfe der spektralen Entmischung analysiert werden, beschrieben. Im darauf folgenden Abschnitt 2.2 wird der Begriff der Reflektanz definiert, da es sich bei allen in dieser Arbeit untersuchten Spektren um Reflektanzspektren handelt. Danach wird in Abschnitt 2.3 das Ziel der spektralen Entmischung definiert. Im Anschluss werden in Abschnitt 2.4 Mechanismen vorgestellt, wie sich Spektren beim Vorhandensein mehrerer Stoffe in einem Pixel zusammensetzen. Darauf aufbauend werden Mischmodelle eingeführt, mit deren Hilfe die spektrale Entmischung durchgeführt werden kann. Danach wird in Abschnitt 2.5 näher auf die Spektrenvariabilität und ihre Ursachen eingegangen. Zuletzt werden in Abschnitt 2.6 Verfahren dargestellt, mit denen die spektrale Entmischung durchgeführt werden kann. Dabei werden nur solche Verfahren vorgestellt, die die hier dargestellten Mischmodelle direkt zur spektralen Entmischung nutzen, und keine, die die spektrale Entmischung mit Hilfe von neuronalen Netzen durchführen. Letztere werden in Kapitel 4 vorgestellt.

2.1 Hyperspektralbilder

In dieser Arbeit bilden Hyperspektralbilder die Datengrundlage für die spektrale Entmischung. Im Gegensatz zu den gängigen RGB-Farbbildern mit drei Farbkanälen (Rot, Grün, Blau) besitzen Hyperspektralbilder eine hohe Anzahl an fein abgetasteten Wellenlängenkanälen. So ergibt sich für jedes Pixel eines Hyperspektralbilds ein diskretes Spektrum. In Abhängigkeit der Anwendung können mehrere hundert Kanäle zum Einsatz kommen, die vom Ultravioletten bis in den Infrarotbereich des

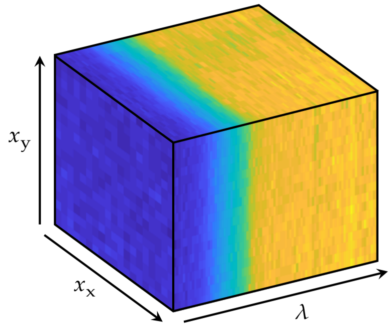


Abbildung 2.1 Oberfläche eines hyperspektralen Datenwürfels in Falschfarben. Jedem Punkt des Würfels wird ein Wert zugeordnet.

elektromagnetischen Spektrums reichen können. Damit steht, verglichen mit RGB-Bildern, wesentlich mehr Information für die Analyse der Materialien in den Pixeln zur Verfügung. Eine häufig verwendete Darstellung für Hyperspektralbilder ist ein Datenwürfel mit zwei Kanten für die örtlichen Dimensionen x_x, x_y und einer dritten Kante für die spektrale Dimension λ (siehe Abbildung 2.1). Die Spektren, die für jedes Pixel vorliegen, können mit den Methoden der Spektroskopie verarbeitet werden. Die skalaren Bilder, die für jeden Wellenlängenkanal vorliegen, können mit den Methoden der Bildverarbeitung ausgewertet werden. Hyperspektralbilder erlauben es darüber hinaus, alle Dimensionen gleichzeitig zu untersuchen (siehe bspw. [4, V4]).

Hyperspektralbilder können mit verschiedenen Verfahren aufgenommen werden. Zum einen wäre das Schnappschussverfahren, welches das vollständige Hyperspektralbild mit einer Aufnahme aufzeichnet, zu nennen. Darüber hinaus existieren scannende Verfahren. Dabei werden Hyperspektralbilder mit einer Zeilenkamera, die alle Wellenlängenkanäle gleichzeitig erfasst, zeilenweise aufgenommen (örtliches Scannen) oder mit Hilfe von Farbfiltern als Grauwertbilder für jeden Wellenlängenkanal nacheinander aufgezeichnet (Scannen entlang der spektralen Dimension) [38]. Je nach Anforderungen (Statik der Szene, Auflösung) kann ein entsprechendes Verfahren gewählt werden.

Die in dieser Arbeit verwendeten hyperspektralen Bilder setzen sich aus Reflektanzspektren zusammen, auf die im nächsten Abschnitt genauer eingegangen wird.

2.2 Reflektanz

Ein Reflektanzspektrum beschreibt die Reflektanz in Abhängigkeit der Wellenlänge. Die Disziplin, die sich mit der Untersuchung von Reflektanzspektren befasst, ist die Reflektanzspektroskopie [136]. Für die spektrale Entmischung spielt in diesem Zusammenhang auch die Theorie des Strahlungstransports eine wichtige Rolle, da damit die Eigenschaften von Partikelmischungen beschrieben werden können [13]. Darauf aufbauend hat sich Hapke [43] sehr ausführlich mit der Reflektanzspektroskopie im Bereich der Fernerkundung auseinandergesetzt.

Die Reflektanz ist dabei die gemessene Größe, die von der Konstellation aus Beleuchtung, zu untersuchendem Objekt und Aufnahmeeinrichtung abhängt [43, Kapitel 8]. Die davon unabhängige Stoffeigenschaft wird als Albedo bezeichnet und beschreibt das Verhältnis zwischen gestreuter und einfallender Strahlungsleistung. Hier wird die über das Volumen gemittelte Einzelstreuungsalbedo $\omega_\lambda \in \mathbb{R}$ eines Partikels eines Stoffes mit

$$\omega_\lambda = \frac{P_\mathrm{S}}{P_\mathrm{S} + P_\mathrm{A}} \tag{2.1}$$

verwendet. Sie ist wellenlängenabhängig und beschreibt das Verhältnis zwischen der Leistung des durch den Partikel gestreuten Lichts $P_\mathrm{S} \in \mathbb{R}$ und der Leistung des auftreffenden Lichts, die wiederum der Summe der Leistung des gestreuten Lichts und der Leistung des absorbierten Lichts $P_\mathrm{A} \in \mathbb{R}$ entspricht. Zu Gunsten der Lesbarkeit wird auf den Wellenlängenindex auf der rechten Seite von (2.1) verzichtet.

Der Zusammenhang zwischen der Einzelstreuungsalbedo und der Reflektanz wird zunächst für die bidirektionale Reflektanz betrachtet. Die bidirektionale Reflektanz $\varGamma_{\mathrm{bi},\lambda} \in \mathbb{R}$ ist das Verhältnis zwischen der von der Oberfläche eines Mediums in eine bestimmte Richtung gestreuten

Strahlung und der einfallenden kollimierten Leistung pro Flächeneinheit, die senkrecht zur Einfallsrichtung steht [43, Kapitel 8 und 10]:

$$\Gamma_{\mathrm{bi},\lambda} = \kappa \frac{\omega_\lambda}{4\pi} \frac{\iota_0}{\iota_0 + \iota} \left\{ \Phi(\phi) + H\Big(\frac{\iota_0}{\kappa}, \omega_\lambda\Big) H\Big(\frac{\iota}{\kappa}, \omega_\lambda\Big) - 1 \right\} . \tag{2.2}$$

Dabei ist $\kappa \in \mathbb{R}$ der Porositätskoeffizient und für $\iota_0, \iota \in \mathbb{R}$ gilt $\iota_0 = \cos(\beta_0)$ bzw. $\iota = \cos(\beta)$. Dabei sind $\beta_0 \in \mathbb{R}$ und $\beta \in \mathbb{R}$ die Winkel zwischen dem einfallenden Licht bzw. dem in Richtung der Kamera gestreuten Licht und dem Normalenvektor der Oberfläche. Die Phasenfunktion $\Phi : \mathbb{R} \to \mathbb{R}$ in Abhängigkeit des Phasenwinkels $\phi \in \mathbb{R}$ zwischen einfallendem und in Richtung der Kamera gestreutem Licht wird zur Beschreibung von Partikeln benötigt, die nicht isotrop streuen [43, Kapitel 8]. Bei der Funktion $H : \mathbb{R}^2 \to \mathbb{R}$ handelt es sich um eine Näherung der H-Funktion nach Subrahmanyan Chandrasekhar, die über eine Integralgleichung definiert ist [13]. Mit ihrer Hilfe wird hier die Mehrfachstreuung beschrieben. In dieser Arbeit wird dafür die Näherung aus [43, Kapitel 8] verwendet, die eine Abweichung von maximal 4 % aufweist:

$$H(\iota, \omega_\lambda) \approx \frac{1 + 2\iota}{1 + 2\iota\sqrt{1 - \omega_\lambda}} . \tag{2.3}$$

In (2.2) wurde der Oppositionseffekt vernachlässigt (siehe [43, Kapitel 9]). Durch diesen erscheinen Objekte bei kleinen Phasenwinkeln heller, da sie ihre eigenen Schatten verdecken. Dies ist für diese Arbeit nicht weiter relevant, da der Winkel zwischen Beleuchtung und Aufnahmeoptik ausreichend groß ist [82].

Für industrielle Anwendungen, bei denen kontrollierte Aufnahmebedingungen herrschen, spielt auch die relative Reflektanz $\Gamma_{\mathrm{rel}} \in \mathbb{R}$ eine wichtige Rolle. Sie beschreibt das Verhältnis zwischen der bidirektionalen Reflektanz des zu untersuchenden Mediums und der eines Reflexionsnormals, die unter den gleichen Bedingungen (gleiche Winkel β_0 und β) bestimmt wird. Das Reflexionsnormal sollte nicht absorbierend sein, isotrop streuen und einen vernachlässigbaren Oppositionseffekt aufweisen. Damit ergibt sich für die relative Reflektanz:

$$\Gamma_{\mathrm{rel},\lambda} = \frac{\kappa_{\mathrm{M}}}{\kappa_{\mathrm{N}}} \omega_\lambda \frac{\Phi(\phi) + H\Big(\frac{\iota_0}{\kappa_{\mathrm{M}}}, \omega_\lambda\Big) H\Big(\frac{\iota}{\kappa_{\mathrm{M}}}, \omega_\lambda\Big) - 1}{H\Big(\frac{\iota_0}{\kappa_{\mathrm{N}}}, 1\Big) H\Big(\frac{\iota}{\kappa_{\mathrm{N}}}, 1\Big)} , \tag{2.4}$$

wobei κ_M und κ_N die Porositätskoeffizienten von Medium und Reflexionsnormal sind. Bei den Spektren in den Datensätzen, die in dieser Arbeit verwendet werden, handelt es sich immer um relative Reflektanzen. Die vorgestellten Methoden für die spektrale Entmischung können jedoch auch für andere Reflektanzspektren verwendetet werden. Nur bei Methoden, die eine Transformation in die Albedo-Domäne erfordern, muss die entsprechende Transformationsvorschrift genutzt werden.

2.3 Ziel der spektralen Entmischung

Die spektrale Entmischung findet immer dann Anwendung, wenn sich ein Pixel eines hyperspektralen Bilds aus den Spektren mehrerer Reinstoffe zusammensetzt und deren Anteile gesucht sind. Diese gemischten Pixel kommen deswegen zustande, weil entweder die räumliche Auflösung des Bildsensors gering ist, wie etwa bei der Fernerkundung, oder, weil die beteiligten Reinstoffe klein sind [60].

Die Reinstoffe sind hier nicht etwa reine Stoffe, die aus nur einer chemischen Verbindung oder einem Element bestehen, sondern hängen von der Anwendung ab. So kann für eine Aufgabenstellung eine gröbere Einteilung ausreichend sein (z. B. Wasser, Grünfläche und Gestein), während an anderer Stelle eine feinere nötig ist (z. B. unterschiedliche Minerale) [10].

Es wird zwischen überwachter und unüberwachter spektraler Entmischung unterschieden. Ziel der unüberwachten spektralen Entmischung ist es, aus einem in Λ Wellenlängenkanälen abgetasteten Spektrum $\boldsymbol{v} \in \mathbb{R}^\Lambda$ die Reinspektren $\mathbf{M} = [\mathbf{m}_1, \dots, \mathbf{m}_P] \in \mathbb{R}^{\Lambda \times P}$ und deren relative Anteile $\mathbf{a} = [a_1, \dots, a_P]^\mathrm{T} \in \mathbb{R}^P$ zu schätzen. Dies ist nur dann möglich, wenn viele Spektren, die aus denselben Reinspektren in unterschiedlichen Anteilen zusammengesetzt sind, verfügbar sind (z. B. bei einem hyperspektralen Bild). Bei der überwachten spektralen Entmischung sind die Reinspektren hingegen bekannt und das Ziel ist die Anteilsschätzung. In dieser Arbeit wird ausschließlich die überwachte spektrale Entmischung betrachtet, da die Reinspektren innerhalb der Trainingsdatensätze für die KNN vorhanden sind. Damit die Schätzung

der relativen Anteile $\hat{\mathbf{a}} = [\hat{a}_1, ..., \hat{a}_P]^\mathrm{T} \in \mathbb{R}^P$ physikalisch plausibel ist, müssen die Nichtnegativitätsbedingung

$$\hat{a}_p \geq 0 \quad \forall p \tag{2.5}$$

und die Normierungsbedingung

$$\sum_{p=1}^{P} \hat{a}_p = 1 \tag{2.6}$$

erfüllt sein.

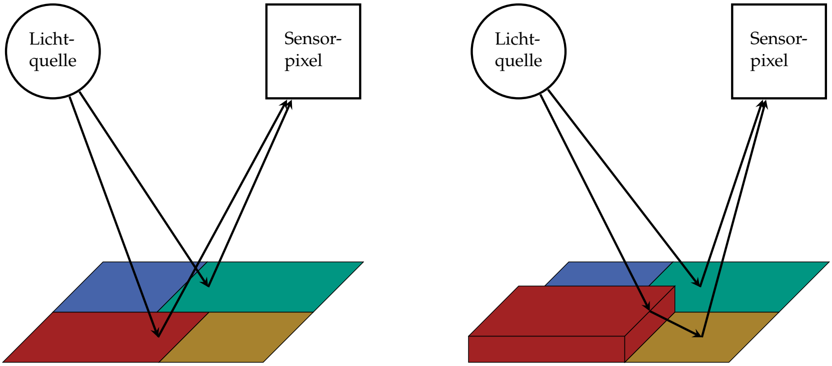
Bei der spektralen Entmischung muss also ein inverses Problem gelöst werden [126]. Dies kann, wie in Kapitel 5, mit einem KNN datenbasiert geschehen oder mit Hilfe von Mischmodellen. Die für diese Arbeit relevanten Mischmodelle werden in Abschnitt 2.4 vorgestellt.

2.4 Mischmodelle

Die Mischspektren unterscheiden sich in Abhängigkeit davon, wie die Reinstoffe gemischt sind. Grundsätzlich muss unterschieden werden, ob die Mischung in einem makroskopischen oder in einem mikroskopischen Maßstab stattfindet.

2.4.1 Mischungen im makroskopischen Maßstab

Mischungen im makroskopischen Maßstab treten dann auf, wenn Photonen, welche ein Sensorpixel einer Kamera erreichen, von unterschiedlichen Reinstoffen dorthin gestreut werden (siehe Abbildung 2.2). Dies ist häufig in der Fernerkundung der Fall, wo die Kamera einen großen Abstand zum untersuchten Objekt hat. Das führt dazu, dass ein Pixel eine größere Fläche abdeckt, die sich wiederum aus Flächen unterschiedlicher Reinstoffe zusammensetzt. Interagiert jedes Photon mit genau einem Reinstoff, bevor es auf den Sensor trifft, entspricht dies Abbildung 2.2(a). Das Mischspektrum ist die mit den Flächenanteilen gewichtete Summe der Spektren der Reinstoffe [43, Kapitel 10]. Das daraus resultierende



(a) Jedes Photon interagiert mit genau einem Reinstoff.

(b) Photonen interagieren auch mit mehreren Reinstoffen.

Abbildung 2.2 Makroskopische Mischungen [22].

Mischmodell wird als lineares Mischmodell (LMM) bezeichnet [15, 22, 23, 60]:

$$\boldsymbol{v} = \sum_{p=1}^{P} a_p \, \mathbf{m}_p = \mathbf{M} \, \mathbf{a} \, . \tag{2.7}$$

Interagieren Photonen jedoch mit mehr als einem Reinstoff, wie es in Abbildung 2.2(b) dargestellt ist, gilt die Annahme des LMMs nicht mehr [49]. Dies wird vor allem durch Mehrfachreflexion bei Oberflächen, die nicht eben sind, verursacht, wie bspw. bei der Fernerkundung von Wäldern oder Bauten in urbanen Regionen [22, 79]. In diesem Fall müssen die Spektren aller Reinstoffe, mit denen ein Photon interagiert, miteinander multipliziert werden. Geläufig sind dabei die bilinearen Mischmodelle, bei denen Photonen, die mit mehr als zwei Reinstoffen interagieren, vernachlässigt werden. Diese Annahme ist zulässig, da bei jeder Interaktion Licht absorbiert und immer weniger Licht gestreut wird.

In dieser Arbeit werden für die spektrale Entmischung das generalisierte bilineare Mischmodell (GBM) [41, 42]

$$\boldsymbol{v} = \sum_{p=1}^{P} \mathbf{m}_p \, a_p + \sum_{p=1}^{P-1} \sum_{q=p+1}^{P} \gamma_{pq} \, a_p \, a_q \, \mathbf{m}_p \odot \mathbf{m}_q \tag{2.8}$$

und das linear-quadratische Mischmodell (LQM) [79]

$$\boldsymbol{v} = \sum_{p=1}^{P} \mathbf{m}_p \, a_p + \sum_{p=1}^{P} \sum_{q=1}^{P} \delta_{pq} \, \mathbf{m}_p \odot \mathbf{m}_q \tag{2.9}$$

verwendet. Dabei sind γ_{pq} und δ_{pq} die Nichtlinearitätskoeffizienten und $\odot$ ist die elementweise Multiplikation. Für $\gamma_{pq} = 1 \; \forall \; p, q$ entspricht das GBM dem Fan-Modell (FM) [30]. Allen Modellen ist gemeinsam, dass zum linearen Teil (erster Summand) ein weiterer Teil hinzukommt, bei dem zwei Spektren elementweise miteinander multipliziert werden. Während der lineare Anteil die Photonen beschreibt, die mit nur einem Reinstoff interagieren, beschreibt der andere Teil diejenigen, die mit zwei Stoffen interagieren. Der Unterschied zwischen GBM, FM und LQM ist, dass bei letzterem auch Spektren mit sich selbst multipliziert werden können, was einer Mehrfachstreuung am gleichen Reinstoff entspricht. Außerdem sind beim LQM die Gewichte der Spektrenprodukte δ_{pq} unabhängig von den Anteilen a_p, während sie beim FM ausschließlich davon abhängen. Beim GBM hängen sie zwar von den Anteilen ab, allerdings sind diese mit einem Koeffizienten γ_{pq} skalierbar.

2.4.2 Mischungen im mikroskopischen Maßstab

Mischungen im mikroskopischen Maßstab liegen vor, wenn ein Stoffgemisch eine homogene Mischung aus Partikeln der Reinstoffe ist. Ein Photon interagiert hier mit vielen Partikeln der beteiligten Reinstoffe (siehe Abbildung 2.3). Die Mischung erfolgt in der Albedo-Domäne.

Zur Modellierung müssen hier die Größen aus (2.2) bzw. (2.4) für die beteiligten Reinstoffe bekannt sein. Im Anschluss daran können die Größen für die Stoffgemische bestimmt und daraus Spektren berechnet werden [43, Kapitel 8]. In der Praxis sind die meisten dieser Größen unbekannt. Sie können bestimmt werden, indem die gleichen Objekte aus unterschiedlichen Winkeln beleuchtet und aufgenommen werden. Liegen ausreichend Messungen vor, können die restlichen Größen daraus geschätzt werden [43, Kapitel 14]. Da auch dies in der Praxis nicht immer möglich oder zu aufwendig ist, werden häufig Vereinfachungen vorgenommen. Damit ist es möglich, die Albedo in Abhängigkeit der

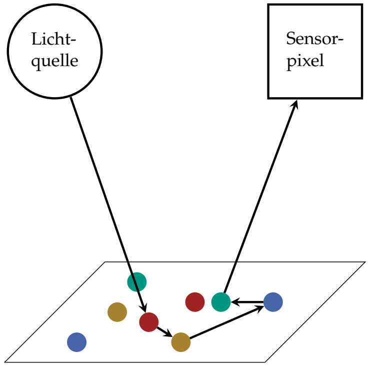


Abbildung 2.3 Mikroskopische Mischung: Photonen interagieren mit vielen Partikeln [22].

Reflektanz analytisch für einen festen Beleuchtungswinkel und einen festen Aufnahmewinkel zu berechnen.

Von Hapke [43, Kapitel 14] wird vorgeschlagen, die Porositätskoeffizienten κ zu 1 zu setzen. Gleichzeitig wird angenommen, dass die Partikel isotrop streuen und damit $\Phi(\phi) = 1 \; \forall \; \phi$ gilt (vgl. [16, 43, 48, 84]). Damit ergibt sich für die relative Reflektanz (Beispiele siehe Abbildung 2.4):

$$\Gamma_{\mathrm{rel},\lambda} = \frac{\omega_\lambda}{\left(1 + 2\iota_0\sqrt{1-\omega_\lambda}\right)\left(1 + 2\iota\sqrt{1-\omega_\lambda}\right)} . \tag{2.10}$$

Häufig wird auch die Umkehrfunktion benötigt, also die Albedo in Abhängigkeit der relativen Reflektanz (mit $\Gamma_{\mathrm{rel}} = \Gamma_{\mathrm{rel},\lambda}$):

$$\omega_\lambda = 1 - \left(\frac{\left[(\iota_0 + \iota)^2 \, \Gamma_{\mathrm{rel}}^2 + (1 + 4\iota_0\iota\Gamma_{\mathrm{rel}})\,(1 - \Gamma_{\mathrm{rel}}) \right]^{\frac{1}{2}} - (\iota_0 + \iota)\,\Gamma_{\mathrm{rel}}}{1 + 4\iota_0\iota\Gamma_{\mathrm{rel}}} \right)^2 . \tag{2.11}$$

Ein Mischspektrum wird nun so modelliert, dass zunächst die Albedo der Mischung und im Anschluss daran mit (2.10) daraus wiederum das Reflektanzspektrum berechnet wird. Die Albedo eines Gemischs

$$\omega_\lambda = \frac{\sum_{p=1}^{P} \frac{\Psi_p}{\rho_p d_p} \omega_{\lambda p}}{\sum_{p=1}^{P} \frac{\Psi_p}{\rho_p d_p}} \tag{2.12}$$

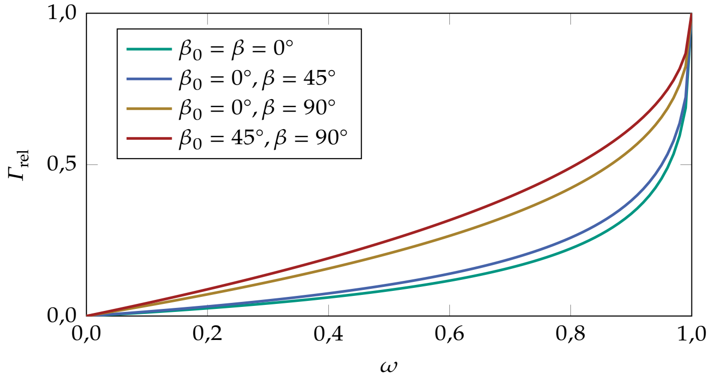


Abbildung 2.4 Relative Reflektanz Γ_{rel} in Abhängigkeit der Albedo ω für unterschiedliche Einfallswinkel β_0 und Streuwinkel β gemäß (2.10).

hängt von den Albedos der beteiligten Reinstoffe $\omega_{\lambda p}$, ihren Massenanteilen $\Psi_p \in \mathbb{R}$, Dichten $\rho_p \in \mathbb{R}$ und Partikeldurchmessern $d_p \in \mathbb{R}$ ab [16, 43]. Sind weitere Größen aus (2.2) der Reinstoffe bekannt, können die entsprechenden Größen, analog zur Albedo, für das Stoffgemisch berechnet werden [43, Kapitel 10].

In der Praxis können, entgegen der vorgestellten Modelle, die Spektren bei gleichen Stoffanteilen mitunter stark variieren. Dies wird als Spektrenvariabilität bezeichnet.

2.5 Spektrenvariabilität

Spektrenvariabilität hat eine große Anzahl unterschiedlichster Ursachen [10]. Eine Darstellung für die Spektrenvariabilität des Reinstoffs Chromoxid-Grün des Farbpulverdatensatzes (siehe Abschnitt 8.1.3) ist in Abbildung 2.5 zu sehen. Effekte durch die Erdatmosphäre spielen in der Fernerkundung, der ursprünglich verbreitetsten Anwendung der spektralen Entmischung, eine wichtige Rolle. Hier sind vor allem die Absorption und die Streuung von Licht durch Moleküle (z. B. Wasserdampf) in der Atmosphäre zu nennen [39]. In dieser Arbeit sind diese jedoch nicht relevant, da der Schwerpunkt auf industriellen Anwendungen liegt.

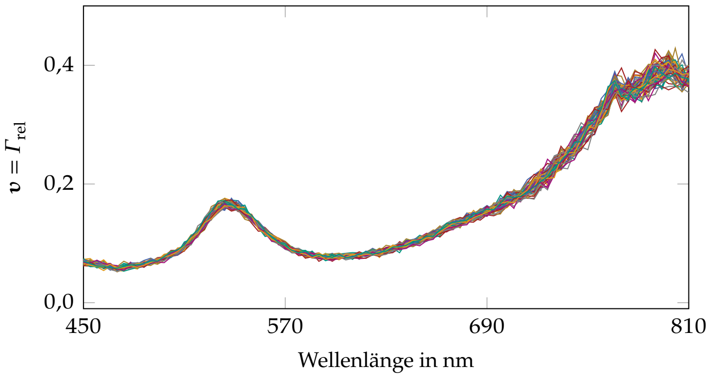


Abbildung 2.5 Beispiele gemessener Spektren des Reinstoffs Chromoxid-Grün des Farbpulverdatensatzes, der in Abschnitt 8.1.3 vorgestellt wird [V7].

Darüber hinaus sind Variationen innerhalb der Reinstoffe zu nennen. Dies tritt vor allem dann auf, wenn die Reinstoffe relativ grob kategorisiert sind. Wenn bspw. die Reinstoffe Grünfläche und bebaute Fläche sind, ist es naheliegend, dass eine große Spektrenvariabilität zu erwarten ist. Auch können die Partikeleigenschaften (vgl. Abschnitte 2.2 und 2.4.2) variieren und eine erhebliche Spektrenvariabilität mit sich bringen.

Weiterhin haben auch die Beleuchtung und die Topografie der betrachteten Oberfläche einen Einfluss auf die Spektren. Sind diese nicht konstant, führen auch sie, aufgrund variierender Winkel zwischen der betrachteten Oberfläche, der Beleuchtung und der Aufnahmeeinrichtung, zu Spektrenvariabilität (vgl. Abschnitt 2.2 und [43]). Je nach Topografie können auch Bereiche im Schatten liegen, was ebenfalls das Spektrum beeinflusst [108].

Die Spektrenvariabilität kann im Mischmodell bzw. bei der spektralen Entmischung berücksichtigt werden [10]. Bei datenbasierten Methoden zur spektralen Entmischung, wie sie in den Kapiteln 5 bis 7 vorgestellt werden, wird die Spektrenvariabilität dann berücksichtigt, wenn sie in den verwendeten Trainingsdaten vorhanden ist. Bei modellbasierter spektraler Entmischung (vgl. 2.6) kann sie durch zusätzliche Parameter

im Mischmodell berücksichtigt werden. Ein wichtiger Vertreter dieser Modelle ist das erweiterte lineare Mischmodell (ELMM) [26, 132]

$$\boldsymbol{v} = \mathbf{M}\, d(\boldsymbol{\phi})\, \mathbf{a}\,, \tag{2.13}$$

wobei $\boldsymbol{\phi} \in \mathbb{R}_+^P$ die zusätzlichen Parameter zur Berücksichtigung der spektralen Entmischung enthält und der Operator d einen Vektor in eine Diagonalmatrix transformiert, wenn das Argument ein Vektor ist. Ist das Argument eine Matrix, wird sie in einen Vektor transformiert, dessen Elemente der Hauptdiagonalen der Matrix entsprechen. Die Elemente in $\boldsymbol{\phi}$ wirken hier als Skalierungsfaktoren für je ein Reinspektrum. Damit können Unterschiede in Beleuchtung und Topografie modelliert werden.

Eine Erweiterung des ELMMs ist das generalisierte lineare Mischmodell (GLMM), bei dem alle Elemente von $\mathbf{M}$ skaliert werden können [54]:

$$\boldsymbol{v} = (\mathbf{M} \odot \boldsymbol{\Phi})\, \mathbf{a}\,. \tag{2.14}$$

Dabei ist $\boldsymbol{\Phi} \in \mathbb{R}_+^{\Lambda \times P}$ die Matrix mit den zusätzlichen Parametern, die für jedes Element von $\mathbf{M}$ einen Skalierungsfaktor darstellen. Damit hat das GLMM wesentlich mehr Freiheitsgrade als das ELMM, wodurch es flexibler ist. Die große Parameterzahl bedeutet jedoch eine große Herausforderung bei der Anwendung für die spektrale Entmischung. Im nächsten Abschnitt werden Verfahren vorgestellt, mit denen alle bisher vorgestellten Mischmodelle für die spektrale Entmischung genutzt werden können.

2.6 Verfahren zur spektralen Entmischung

Bei der modellbasierten überwachten spektralen Entmischung werden $\hat{\mathbf{a}}$ und die übrigen Modellparameter solange angepasst, bis der Fehler zwischen dem gemessenen Spektrum und dem modellierten Spektrum möglichst klein ist. Dabei müssen stets die Nebenbedingungen (2.5) und (2.6) eingehalten werden. Meist wird hier der mittlere quadratische Fehler

$$\hat{\boldsymbol{\theta}} = \arg\min_{\boldsymbol{\theta}} \|\boldsymbol{v} - \mathbf{f}(\mathbf{M}, \boldsymbol{\theta})\|_2^2 \tag{2.15}$$

verwendet, wobei die Funktion $\mathbf{f}$ stellvertretend für ein beliebiges Mischmodell steht, während der Vektor $\boldsymbol{\theta}$ für alle bei diesem Mischmodell enthaltenen Parameter steht und $\hat{\boldsymbol{\theta}}$ für deren Schätzung.

2.6.1 Lineare Mischmodelle

Beim LMM wird das Optimierungsproblem (2.15) zu dem quadratischen Optimierungsproblem

$$\hat{\mathbf{a}} = \arg\min_{\mathbf{a}} \|\boldsymbol{v} - \mathbf{M}\,\mathbf{a}\|_2^2 \tag{2.16}$$

mit den Nebenbedingungen (2.5) und (2.6). Dieses Optimierungsproblem ist stets konvex, da die Hesse-Matrix $\mathbf{M}^{\mathrm{T}}\mathbf{M}$ positiv definit ist. Damit ist jedes lokale Minimum auch ein globales Minimum. Ein Verfahren, welches bei der spektralen Entmischung gängig ist und die Nebenbedingungen (2.5) und (2.6) miteinbezieht, ist das *Fully-Constrained-Least-Squares*-Verfahren (FCLS) [47]. Dabei wird die Lagrange-Funktion

$$L(\mathbf{a}, l) = \|\boldsymbol{v} - \mathbf{M}\,\mathbf{a}\|_2^2 - l\left(\sum_{p=1}^{P} a_p - 1\right) \tag{2.17}$$

mit dem Lagrange-Multiplikator $l \in \mathbb{R}$ minimiert. Es handelt sich um ein iteratives Verfahren, bei dem negative $\mathbf{a}_p$ und die korrespondierenden Reinspektren aus $\mathbf{M}$ entfernt werden.

Bei den vorgestellten Erweiterungen ELMM (2.13) und GLMM (2.14) des LMMs wird (2.15) zu einem nichtkonvexen Optimierungsproblem. Drumetz et al. [27] verwenden ein iteratives Optimierungsverfahren für das ELMM, welches abwechselnd $\boldsymbol{\phi}$ und $\mathbf{a}$ trainiert. Die beiden Teilprobleme sind dabei konvex und es wird so lange optimiert, bis ein Abbruchkriterium erreicht wird. Darauf baut das Optimierungsverfahren für das GLMM auf [54]. Dabei werden die Parameter ebenfalls abwechselnd optimiert. Ein wichtiger Unterschied besteht darin, dass im Gegensatz zum Optimierungsverfahren für das ELMM das Optimierungsproblem nicht für einzelne Spektren separierbar ist, sondern immer das ganze Hyperspektralbild (der ganze Datensatz) optimiert wird. Dies ist aufgrund der vielen Freiheitsgrade, die das GLMM bietet, notwendig.

2.6.2 Nichtlineare Mischmodelle

Bei den nichtlinearen Mischmodellen GBM und LQM handelt es sich ebenfalls um nichtkonvexe Optimierungsprobleme. Um diese zu lösen,

werden unterschiedliche Ansätze vorgeschlagen [42]. In dieser Arbeit wird dazu das Gradientenabstiegsverfahren verwendet. Dies kommt auch beim Training von KNN zum Einsatz und wird in Abschnitt 3.2.3 genauer erläutert.

Auch mit den physikalisch motivierten Überlegungen von Hapke (2.12) kann eine spektrale Entmischung durchgeführt werden. Dazu wird mit unter den im Abschnitt 2.4.2 genannten Annahmen und für feste Winkel β, β_0 der Zusammenhang (2.11) genutzt, um die Reflektanzspektren zunächst in die Albedo-Domäne zu transformieren. Unter den Annahmen, dass der mittlere Partikeldurchmesser und die mittlere Dichte der beteiligten Reinstoffe ungefähr gleich groß sind, kann (2.12) weiter vereinfacht werden:

$$\omega_\lambda = \sum_{p=1}^{P} a_p \, \omega_{\lambda p} \, . \tag{2.18}$$

Dabei sind die Anteile a_p die relativen Masseanteile der Reinstoffe. Da aber eine identische Dichte für alle Reinstoffe angenommen wird, sind diese Anteile auch weiterhin Volumenanteile, wie bei den anderen Mischmodellen auch. In Vektorschreibweise wird (2.18) zu

$$\boldsymbol{\omega} = \sum_{p=1}^{P} a_p \, \boldsymbol{\omega}_p \, , \tag{2.19}$$

worauf die Methoden zur spektralen Entmischung für das LMM aus Abschnitt 2.6.1 angewendet werden können. Auch wenn viele Annahmen getroffen werden müssen, um dieses Resultat zu erhalten, so können damit in der Praxis gute Ergebnisse erzielt werden [16, 48, 84].

3 Künstliche neuronale Netze

Dieses Kapitel beinhaltet die Grundlagen und Definitionen zu KNN, auf denen die weiteren Kapitel aufbauen. Künstliche neuronale Netze gehören zu den Methoden des maschinellen Lernens, welches wiederum ein Teilbereich der künstlichen Intelligenz ist. Sie wurden bereits 1943 erstmals als Verknüpfungen elementarer Einheiten von McCulloch und Pitts beschrieben [77].

Hier werden in Abschnitt 3.1 zunächst die Grundlagen des maschinellen Lernens dargestellt, bevor in Abschnitt 3.2 auf die Grundlagen von KNN eingegangen wird. In den Abschnitten 3.3, 3.4 und 3.5 werden mit Faltungsnetzen, GAN und *Autoencodern* drei für diese Arbeit wichtige Arten von KNN vorgestellt.

3.1 Grundlagen des maschinellen Lernens

Bereits 1959 beschreibt *Arthur L. Samuel* maschinelles Lernen als Fähigkeit von Computern, ohne explizite Programmierung zu lernen [111]. Diese Grundidee steht weiterhin im Zentrum des maschinellen Lernens. Modelle werden dabei nicht auf Basis physikalischer Grundlagen bzw. der Erfahrung von Experten aufgestellt. Stattdessen lernt ein Modell Zusammenhänge basierend auf seiner eigenen Erfahrung in Form von Beispieldaten [63]. Nach dem (erfolgreichen) Lernprozess kann es seine Erfahrung auf unbekannte Daten anwenden.

Auch wenn das Konzept nicht neu ist, wurden vor allem in den letzten Jahren große Erfolge damit erzielt. Das liegt vor allem daran, dass zum einen die Rechenleistung immer weiter steigt und zum anderen Speicher immer günstiger wird. Besonders die Einführung von *general-purpose computing on graphics processing units* Anfang des Jahrtausends brachte einen großen Schub in Sachen Rechenleistung. Damit war es möglich, durch sogenannte *single-instruction-multiple-data*-Operationen ein hohes

Maß an Parallelisierung bei der Verarbeitung von Gleitkommazahlen auf Grafikkarten zu erreichen. Dies beschleunigt auch das Training und die Inferenz von KNN signifikant [14]. Darüber hinaus stieg auch die Anzahl an verfügbaren Daten. Letzteres hängt stark von der Art der Daten und der Anwendung ab. Während allein die Tatsache, dass viele Menschen ein Mobiltelefon mit Kamera besitzen, die Anzahl an digitalen Farbbildern extrem erhöht hat, müssen Daten für spezielle Anwendungen in der Industrie meist erst aufwendig erfasst und gekennzeichnet werden. Vor allem dort, wo eine Modellierung durch den Menschen besonders komplex ist, wie bspw. in der Bildverarbeitung oder der Verarbeitung von Sprache, können große Erfolge erzielt werden [19, 68, 123].

Wie eingangs erwähnt, beziehen Methoden des maschinellen Lernens ihre Erfahrung aus Daten. Dabei wird zwischen Ein- und Ausgangsdaten unterschieden. Beim überwachten Lernen werden sowohl die Eingangsdaten als auch die zugehörigen Ausgangsdaten verwendet. Ziel ist es, den Zusammenhang zwischen Ein- und Ausgangsdaten zu lernen, sodass der Algorithmus in der Lage ist, für unbekannte Eingangsdaten sinnvolle Ausgangswerte zu ermitteln [130]. Überwachtes Lernen ist die Art des maschinellen Lernens, die in dieser Arbeit hauptsächlich verwendet wird. Daneben gibt es das unüberwachte Lernen, welches keine Ausgangsdaten benötigt, sondern mit Hilfe von Eingangsdaten ein statistisches Modell erlernen kann. Darüber hinaus existiert das bestärkende Lernen (engl. *reinforcement learning*). Bei letzterem geht es darum, eigenständig eine Strategie zu entwickeln, die ein bestimmtes Gütemaß (Belohnung) maximiert [122].

3.1.1 Aufgabenstellungen

Methoden des maschinellen Lernens können für unterschiedliche Aufgaben herangezogen werden [35, Kapitel 5]. Für diese Arbeit sind vor allem die Klassifikation und die Regression relevant. Das Ziel bei der Klassifikation ist die Zuordnung eines Eingangsdatenpunkts zu einer Klasse oder Kategorie. Die Klassen sind diskret und es existiert eine endliche Menge an Klassen. Ein Beispiel für eine Klassifikationsaufgabe wäre die Zuordnung von Reinspektren zu den dazugehörigen Reinstoffen.

Bei der Regression hingegen wird einem Eingangsdatenpunkt ein reeller Zahlenwert oder ein Vektor mit reellen Zahlen zugeordnet:

$$f : \mathbb{R}^J \to \mathbb{R}^K \,, \tag{3.1}$$

wobei $J \in \mathbb{N}$ und $K \in \mathbb{N}$ für die Dimension des Eingangs- bzw. Ausgangsdatenpunkts stehen. Bei der zentralen Aufgabenstellung dieser Arbeit, der spektralen Entmischung, handelt es sich um ein Regressionsproblem. Hier wird einem Spektrum ein Anteilsvektor zugeordnet.

3.1.2 Modellparameter

Die Methoden des maschinellen Lernens können in parametrische und nichtparametrische Methoden eingeteilt werden. Da in dieser Arbeit keine nichtparametrischen Methoden behandelt werden, wird auf diese hier nicht weiter eingegangen.

Bei parametrischen Methoden muss zunächst eine von Parametern $\boldsymbol{\theta}$ abhängige Annahme getroffen werden, wie Ein- und Ausgangswerte zusammenhängen. Der Zusammenhang zwischen Ein- und Ausgangswerten kann als parametrisierte Funktion $f_{\boldsymbol{\theta}}$ dargestellt werden. Beispielsweise kann für eine Regression ein Polynom einer bestimmten Ordnung angenommen werden, wobei die Koeffizienten die zu bestimmenden Parameter sind. Die Parameter werden dann mit Hilfe von Daten bestimmt.

3.1.3 Daten und Augmentierung

Beim maschinellen Lernen spielt die Qualität der Daten eine entscheidende Rolle. Die Beispieldaten, mit denen die Zusammenhänge erlernt werden, heißen Trainingsdaten, der Lernprozess wird als Training bezeichnet. Beim überwachten Lernen handelt es sich bei den Trainingsdaten um Paare aus Ein- und Ausgangsdaten $\{\mathbf{x}_n, \mathbf{y}_n^*\}$ mit $n \in \{1, \dots, N\}$. Dabei ist $\mathbf{x}_n \in \mathbb{R}^J$ ein Eingangsdatenpunkt und $\mathbf{y}_n^* \in \mathbb{R}^K$ der dazugehörige wahre Ausgang (engl. *ground truth*). Ein Trainingsdatensatz $\mathcal{X} = \left\{\{\mathbf{x}_1, \mathbf{y}_1^*\}, \dots, \{\mathbf{x}_N, \mathbf{y}_N^*\}\right\}$ besteht aus vielen dieser Datenpunktpaare. Diese sollten einen großen Bereich der möglichen Ein- und Ausgangswerte abdecken, um repräsentativ zu sein.

Da Trainingsdaten oft nur sehr aufwendig gewonnen werden können und deswegen Trainingsdatensätze für spezielle Aufgaben (z. B. im industriellen Umfeld) recht klein ausfallen, haben sich Methoden etabliert, die diese künstlich erweitern. Der Prozess der Datensatzerweiterung wird als Augmentierung bezeichnet. In der Bildverarbeitung geht es dabei vor allem darum, bezüglich bestimmter Eigenschaften wie Rotation oder Skalierung Invarianzen zu erzeugen [113]. Im Zusammenhang mit KNN spielt die Augmentierung eine wichtige Rolle und so wurden Verfahren entwickelt, die systematisch nach guten Augmentierungsstrategien suchen [17, 81]. Auch in dieser Arbeit wird Augmentierung benutzt, um zusätzliche Datenpunkte zu erzeugen, die zwischen den Anteilsvektoren der vorhandenen Datenpunkte liegen (siehe Abschnitt 2.3). Damit soll die Regression verbessert werden, sodass sie trotz endlichem Trainingsdatensatz für Spektren mit zugehörigen unbekannten Anteilsvektoren sinnvolle Ergebnisse liefert.

Viele Methoden des maschinellen Lernens besitzen Konfigurationsmöglichkeiten. Die zugehörigen Größen werden als Hyperparameter bezeichnet. Beispielsweise ist die Anzahl an Schichten in einem neuronalen Netz (siehe 3.2) ein Hyperparameter. Zur Bestimmung guter Hyperparameter wird ein Teil der Daten nicht für das Training verwendet, sondern für die Validierung aufgespart. Der sogenannte Validierungsdatensatz wird nach jedem Trainingsprozess mit unterschiedlichen Hyperparametern ausgewertet. In einem iterativen Prozess können so gute Hyperparameter gefunden werden.

Soll eine Bewertung des trainierten Modells stattfinden, werden weitere Datenpunkte benötigt. Diese werden als Testdaten bezeichnet und dürfen beim Trainingsprozess nicht verwendet werden. So kann sichergestellt werden, dass das trainierte Modell nicht den Trainingsdatensatz auswendig gelernt hat, sondern tatsächlich abstrakte Zusammenhänge gelernt wurden. Nur dann war der Trainingsprozess erfolgreich und das Modell kann für unbekannte Eingangsdaten sinnvolle Ausgangsdaten liefern.

3.1.4 Verlustfunktion

Um die Modellparameter aus Abschnitt 3.1.2 zu bestimmen, wird eine Verlustfunktion (engl. *loss function*) oder Kostenfunktion $l_1 : \mathbb{R}^{2K} \rightarrow \mathbb{R}$

verwendet. Diese vergleicht den Ausgang des Modells $\mathbf{y}_n = f_{\boldsymbol{\theta}}(\mathbf{x}_n) \in \mathbb{R}^K$ mit dem dazugehörigen wahren Wert $\mathbf{y}_n^*$ und berechnet mit Hilfe eines Distanzmaßes einen einzelnen Wert. Dies wird in den meisten Fällen nicht für einen einzelnen Datenpunkt, sondern für N Datenpunkte gleichzeitig durchgeführt. Damit die Verlustfunktion dennoch in einem Skalar resultiert, wird meist gemittelt. Dies entspricht einer Verlustfunktion $\ell : \mathbb{R}^{2KN} \to \mathbb{R}$ mit

$$\ell = \frac{1}{N} \sum_{n=1}^{N} \ell_1(\mathbf{y}_n, \mathbf{y}_n^*) = \frac{1}{N} \sum_{n=1}^{N} \ell_1(f_{\boldsymbol{\theta}}(\mathbf{x}_n), \mathbf{y}_n^*) . \tag{3.2}$$

Es gilt, die Verlustfunktion während des Trainings zu minimieren, indem die Parameter $\boldsymbol{\theta}$ des Modells angepasst werden. Je nach Aufgabenstellung sind unterschiedliche Verlustfunktionen sinnvoll. Für eine Regression, wie die spektrale Entmischung, wird häufig der mittlere quadratische Fehler (engl. *mean squared error*, MSE)

$$\ell_{\mathrm{MSE}} = \frac{1}{NK} \sum_{n=1}^{N} \left\| \mathbf{y}_n - \mathbf{y}_n^* \right\|_2^2 \tag{3.3}$$

verwendet, der mit der Energie des Fehlers korrespondiert. Der MSE dient ebenso als Gütekriterium bei der Bewertung des Testdurchgangs.

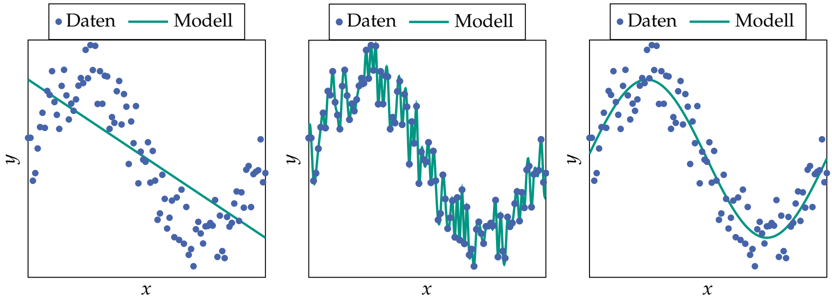
Eine weitere wichtige Verlustfunktion in dieser Arbeit ist die logistische Verlustfunktion

$$\ell_{\log} = \frac{1}{NK} \sum_{n=1}^{N} \sum_{k=1}^{K} y_{nk}^* \log(y_{nk}) + (1 - y_{nk}^*) \log(1 - y_{nk}) , \tag{3.4}$$

die nur für den Wertebereich zwischen 0 und 1 definiert ist. Sie wird häufig im Zusammenhang mit der logistischen Funktion (3.7) (Aktivierungsfunktion bei KNN) verwendet.

3.1.5 Überanpassung und Unteranpassung

Das wichtigste Ziel beim maschinellen Lernen ist, dass das trainierte Modell nicht nur mit den Trainingsdaten gute Ergebnisse erzielt, sondern mit Daten, die es zuvor nie gesehen hat. Diese Fähigkeit kann mit Hilfe



(a) Unteranpassung. **(b)** Überanpassung. **(c)** Gute Generalisierung.

Abbildung 3.1 Illustration von Unteranpassung, Überanpassung und einer guten Generalisierung am Beispiel einer Regression mit eindimensionaler Ein- und Ausgangsgröße.

des Testdatensatzes untersucht werden (siehe Abschnitt 3.1.3). Die Eigenschaft, gute Ergebnisse mit bisher unbekannten Daten zu erzielen, wird als Generalisierung bezeichnet [35, Kapitel 5]. Es geht also darum, möglichst gut Zusammenhänge und nicht den Trainingsdatensatz auswendig zu lernen. Die Probleme, die dabei entstehen können, sind Unteranpassung (engl. *underfitting*) und Überanpassung (engl. *overfitting*).

Unteranpassung tritt dann auf, wenn das Modell nicht in der Lage ist, die Zusammenhänge aus dem Trainingsdatensatz zu lernen. Die Ursache dafür ist, dass das Modell bzw. die parametrisierte Funktion (siehe Abschnitt 3.1.2) falsch gewählt wird und die Zusammenhänge in den Daten für keine Parameterkombination ausreichend gut abgebildet werden können. Ein Beispiel dafür wird in Abbildung 3.1(a) illustriert. Der sinusförmige Verlauf der Datenpunkte kann nur unzureichend mit einer linearen Funktion approximiert werden. Ein weiterer Grund für eine Unteranpassung ist eine zu kurze Trainingsdauer.

Um dem entgegenzuwirken, muss ein komplexeres Modell bzw. eine komplexere Funktion angenommen werden. Dies wird auch als Erhöhung der Kapazität des Modells bezeichnet [44]. Bei einer hohen Kapazität besteht die Gefahr, dass Überanpassung auftritt. Das bedeutet, dass die Datenpunkte des Trainingsdatensatzes besonders gut durch das Modell abgebildet werden, unbekannte Datenpunkte jedoch zu schlechten Ergebnissen führen, d. h. die Fähigkeit der Generalisierung nimmt ab. Ein Beispiel dafür wird in Abbildung 3.1(b) illustriert. Hier verläuft die

Funktion durch alle Datenpunkte, die jedoch verrauscht sind. Da das Rauschen zufällig auftritt, ist es unwahrscheinlich, dass die Funktion für unbekannte Datenpunkte passend ist. Überanpassung lässt sich vermeiden, indem die Kapazität des Modells nicht zu groß gewählt wird und das Training rechtzeitig beendet wird. Hierzu dient die Überprüfung mit dem Validierungsdatensatz. Während der Fehler des Trainingsdatensatzes immer weiter sinkt, sofern die Kapazität ausreichend ist, steigt der Fehler des Validierungsdatensatzes ab einem bestimmten Punkt wieder an. Ein Trainingsdatensatz bestehend aus Datenpunkten, die dazu eine hohe Varianz aufweisen, wirkt der Überanpassung entgegen, da eine extrem hohe Kapazität nötig wäre, um sich an diese überanzupassen.

Vor allem bei komplexeren Problemen, bei denen ein geeignetes Modell nicht einfach ersichtlich ist, ist es eine große Herausforderung, ein Modell zu wählen, das eine geeignete Kapazität aufweist. Gerade bei kleineren Datenmengen im Trainingsdatensatz tritt sonst schnell Überanpassung auf. In Abbildung 3.1(c) ist ein Beispiel für eine gute Generalisierung zu sehen. Hier wird das Rauschen ignoriert, während der Zusammenhang zwischen Ein- und Ausgang gut abgebildet wird.

Im Folgenden werden die Grundlagen zu KNN dargestellt, die in dieser Arbeit als Methode des maschinellen Lernens verwendet werden und für die die bisher vorgestellten Grundlagen gelten.

3.2 Grundlagen künstlicher neuronaler Netze

Künstliche neuronale Netze gehören zu den parametrischen Methoden des maschinellen Lernens und entsprechen daher einer parametrisierten Funktion $f_{\boldsymbol{\theta}}$. Beim Aufstellen dieser Funktion fließen jedoch keine Annahmen über den Zusammenhang zwischen Ein- und Ausgangsgrößen direkt ein (vgl. Abschnitt 3.1.2). Stattdessen wird ein KNN entworfen, das einer Funktion mit sehr vielen Parametern und damit sehr vielen Freiheitsgraden entspricht. Beim Entwurf des KNNs ist es jedoch durchaus hilfreich, die Menge und Beschaffenheit der vorhandenen Trainingsdaten zu berücksichtigen.

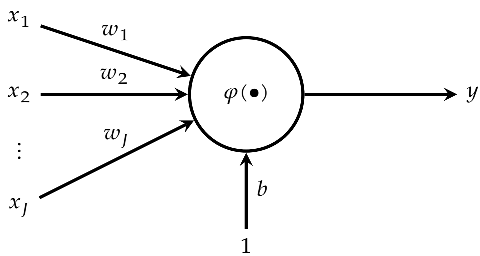


Abbildung 3.2 Künstliches Neuron.

3.2.1 Multilayer Perceptron

Künstliche neuronale Netze bestehen aus künstlichen Neuronen. Ein solches ist in Abbildung 3.2 zu sehen. Es stellt eine Beziehung zwischen mehreren Eingangswerten $\mathbf{x} = [x_1, \ldots, x_J]^\mathrm{T} \in \mathbb{R}^J$ und einem Ausgangswert $y \in \mathbb{R}$ her:

$$y = \varphi\left(\sum_{j=1}^{J} w_j x_j + b\right) = \varphi\left(\mathbf{w}^\mathrm{T}\mathbf{x} + b\right) . \tag{3.5}$$

Dabei sind $\mathbf{w} = [w_1, \ldots, w_J]^\mathrm{T} \in \mathbb{R}^J$ die Gewichte des Neurons, $b \in \mathbb{R}$ der Schwellenwert (engl. *bias*) und $\varphi : \mathbb{R} \to \mathbb{R}$ die Aktivierungsfunktion.

Die Neuronen können prinzipiell beliebig angeordnet sein. Diese Arbeit beschränkt sich jedoch auf vorwärts gerichtete (engl. *feedforward*) Netze. Diese besitzen keine Schleifen und die Neuronen sind in Schichten angeordnet. Jedes Neuron einer Schicht ist mit allen Neuronen der vorhergehenden Schicht und allen Neuronen der nachfolgenden Schicht verbunden. Eine Schicht kann mit

$$\mathbf{y} = \boldsymbol{\varphi}\left(\mathbf{W}\mathbf{x} + \mathbf{b}\right) \tag{3.6}$$

beschrieben werden. Dabei sind $\mathbf{y} = [y_1, \ldots, y_K]^\mathrm{T} \in \mathbb{R}^K$ der Ausgang, $\mathbf{W} = [\mathbf{w}_1, \ldots, \mathbf{w}_K]^\mathrm{T} \in \mathbb{R}^{KJ}$ die Gewichte und $\mathbf{b} = [b_1, \ldots, b_K]^\mathrm{T} \in \mathbb{R}^K$ die Schwellenwerte der Schicht. Die Aktivierungsfunktion $\boldsymbol{\varphi} : \mathbb{R}^K \to \mathbb{R}^K$ wird elementweise angewandt.

Sind die Gewichte $\mathbf{W}$ beliebig, handelt es sich um eine vollständig verbundene Schicht. Ein KNN, das nur aus vollständig verbundenen

Schichten besteht, wird als *Multilayer Perceptron* bezeichnet [109]. Die erste Schicht ist dabei die Eingangsschicht, die letzte die Ausgangsschicht. Alle Schichten dazwischen werden als verborgene Schichten (engl. *hidden layers*) bezeichnet.

3.2.2 Aktivierungsfunktion

Künstliche neuronale Netze müssen nichtlineare Aktivierungsfunktionen enthalten. Nur dann sind KNN universelle Approximatoren [93]. Das universelle Approximationstheorem sagt aus, dass jede stetige Funktion durch ein KNN beliebig genau approximiert werden kann, wenn es entsprechend ausgelegt wird. Ohne nichtlineare Aktivierungsfunktion wären KNN auf die Approximation linearer Zusammenhänge limitiert [35, Kapitel 6], da eine Verkettung linearer Operationen in einer linearen Operation resultiert.

Es existieren viele Aktivierungsfunktionen, deren Wahl Teil des Entwurfs des KNNs ist. Theoretisch kann für jedes Neuron die Aktivierungsfunktion separat festgelegt werden, in der Praxis wird jedoch meist die gleiche Aktivierungsfunktion für alle Neuronen festgelegt. Eine Ausnahme ist die Ausgangsschicht, die je nach Zielsetzung eine andere Aktivierungsfunktion hat. In dieser Arbeit werden die logistische Funktion

$$\varphi_{\log}(x) = \frac{1}{1 + \mathrm{e}^{-x}} \tag{3.7}$$

und die ReLU-Funktion (engl. *rectified linear unit*, [45])

$$\varphi_{\mathrm{ReLU}}(x) = \max(0, x) \tag{3.8}$$

verwendet (siehe Abbildung 3.3). Die logistische Funktion, oft auch als Sigmoid-Funktion bezeichnet, hat die Vorteile, dass sie stetig differenzierbar ist und im Bereich zwischen 0 und 1 liegt. Letzteres ist vor allem für die Ausgangsschicht interessant, wenn der Wertebereich der Ausgangsgrößen genau in diesem Bereich liegt, wie es auch in dieser Arbeit der Fall ist (Spektren nach Weißbildabgleich und Anteile). Nachteilig sind der vergleichsweise hohe Rechenaufwand und sehr kleine Gradienten für betragsmäßig große Eingangsgrößen. Die Gradienten sind beim Training wichtig und beeinflussen die Dauer maßgeblich. Zu kleine Gradienten haben eine hohe Trainingsdauer zur Folge. Darüber hinaus steigt

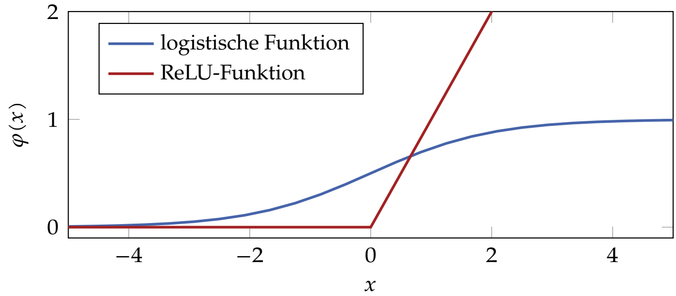


Abbildung 3.3 Aktivierungsfunktionen.

die Wahrscheinlichkeit, dass die Optimierung der Parameter in einem lokalen Minimum endet, welches schlechte Ergebnisse liefert. Außer der Aktivierungsfunktion haben jedoch noch andere Faktoren Einfluss auf die Größe der Gradienten (siehe Abschnitt 3.2.3).

Die ReLU-Funktion hat einen vergleichsweise geringen Rechenaufwand, außerdem ist der Gradient für positive Werte stets 1. Nachteilig ist jedoch, dass der Gradient für negative Eingänge den Wert 0 aufweist, wodurch das Training stagnieren kann: Ist bei einem Neuron einmal ein negativer Ausgangswert erreicht, so kann dieser Zustand nicht mehr verlassen werden, wenn das für alle beim Training verwendeten Datenpunkte der Fall ist. Daher empfiehlt es sich, keine zu hohe Lernrate (siehe Abschnitt 3.2.3) zu nutzen, wenn die ReLU-Funktion als Aktivierungsfunktion verwendet wird. Ein heterogener Trainingsdatensatz wirkt diesem Problem ebenfalls entgegen.

3.2.3 Training

Beim Training eines KNNs wird ein Optimierungsproblem gelöst, bei dem die Verlustfunktion (3.2) minimiert wird, indem die Parameter $\boldsymbol{\theta}$ angepasst werden. Dabei handelt es sich im Allgemeinen um ein nichtkonvexes Optimierungsproblem, weshalb nur lokale Minima gefunden werden können. In der Praxis stellt dies kein Problem dar. Im Gegenteil, im globalen Minimum bezogen auf einen Trainingsdatensatz liegt sehr wahrscheinlich Überanpassung vor (vgl. Abschnitt 3.1.5). Mit ge-

eigneten Methoden ist es möglich, falls die übrigen Voraussetzungen stimmen (guter Datensatz, sinnvolle Netzarchitektur), ein „gutes“ lokales Minimum zu erreichen. Die gängigen Verfahren zur Optimierung neuronaler Netze basieren auf dem Gradienten der Verlustfunktion bezüglich des Parametervektors $\boldsymbol{\theta}$. Letzterer setzt sich aus den Gewichten $\mathbf{W}$ und Schwellenwerten $\mathbf{b}$ aller Schichten des KNNs zusammen. Um den Gradienten effizient zu berechnen, wird die Fehlerrückführung (engl. *backpropagation*) verwendet [110].

3.2.3.1 Backpropagation

An dieser Stelle werden die Grundlagen im Sinne der Lesbarkeit für einen Datenpunkt dargestellt. In der Praxis wird die Fehlerrückführung jedoch meist für viele Datenpunkte oder den ganzen Trainingsdatensatz gleichzeitig durchgeführt (siehe Abschnitt 3.2.3.3). Bei mehreren Datenpunkten werden die entsprechenden Gradienten abhängig von der Implementierung gemittelt oder aufsummiert. Der resultierende Gradient hat in beiden Fällen die gleiche Richtung. Es ändert sich nur der Betrag, der durch die Lernrate angepasst werden kann (siehe Abschnitt 3.2.3.2).

Ziel der Fehlerrückführung ist die Bestimmung des Gradienten

$$\boldsymbol{\vartheta} = \nabla_{\boldsymbol{\theta}} \ell(f_{\boldsymbol{\theta}}(\mathbf{x}), \mathbf{y}^*) \,. \tag{3.9}$$

Dies ist bei einem *Multilayer Perceptron* äquivalent zur Bestimmung der Gradienten der Verlustfunktion bezüglich aller im KNN vorkommenden Gewichte $\mathbf{W}$ und Schwellenwerte $\mathbf{b}$. Die Berechnung erfolgt mit Hilfe der Kettenregel, wonach für die Ableitung einer verketteten Funktion gilt:

$$(f_2 \circ f_1)'(x) = f_2'(f_1(x)) \cdot f_1'(x) \,. \tag{3.10}$$

Übertragen auf künstliche neuronale Netze ergibt sich bspw. für eine vollständig verbundene Schicht (3.6), wobei das Argument von ℓ zu Gunsten der Lesbarkeit nicht dargestellt wird:

$$\nabla_{\mathbf{x}} \ell = \mathbf{W}^{\mathrm{T}} \boldsymbol{\varphi}'(\mathbf{W}\mathbf{x} + \mathbf{b}) \odot \nabla_{\mathbf{y}} \ell \tag{3.11}$$

$$\nabla_{\mathbf{b}} \ell = \boldsymbol{\varphi}'(\mathbf{W}\mathbf{x} + \mathbf{b}) \odot \nabla_{\mathbf{y}} \ell \tag{3.12}$$

$$\nabla_{\mathbf{W}} \ell = \nabla_{\mathbf{y}} \ell \odot \boldsymbol{\varphi}'(\mathbf{W}\mathbf{x} + \mathbf{b}) \mathbf{x}^{\mathrm{T}} \,. \tag{3.13}$$

Bei der Ableitung der Aktivierungsfunktion nach ihrem Argument würde eigentlich eine Jacobi-Matrix herauskommen. Diese hat jedoch nur Einträge in der Hauptdiagonalen, weshalb obige Formulierung mit $\odot$ korrekt ist. Mit Hilfe von $\nabla_{\mathbf{b}}\ell$ und $\nabla_{\mathbf{W}}\ell$ können nun mit einem Optimierer (siehe Abschnitt 3.2.3.2) direkt die neuen Parameter berechnet werden. Während $\nabla_{\mathbf{x}}\ell$ als $\nabla_{\mathbf{y}}\ell$ an die vorherige Schicht weitergegeben wird, die wiederum ihrerseits die Gradienten (3.11) bis (3.13) berechnet, entspricht $\nabla_{\mathbf{y}}\ell$ dem Gradienten $\nabla_{\mathbf{x}}\ell$ der nachfolgenden Schicht. In der letzten Schicht ist $\nabla_{\mathbf{y}}\ell$ die Ableitung der Verlustfunktion. Sind alle Gradienten der Parameter berechnet, können mit Hilfe eines Optimierers die Parameter aktualisiert werden.

3.2.3.2 Optimierer

Die einfachste Möglichkeit, um die Parameter anzupassen, ist in Richtung des negativen Gradienten zu gehen [12, 73]:

$$\boldsymbol{\theta}_t = \boldsymbol{\theta}_{t-1} - \alpha \boldsymbol{\vartheta}_t \,. \tag{3.14}$$

Dabei ist $t \in \mathbb{N}$ der Trainingsschritt, $\boldsymbol{\theta}_0$ die Initialisierung der Parameter und $\alpha \in \mathbb{R}^+$ die Lernrate. Letztere ist ein Hyperparameter beim Training des KNNs und kann entweder konstant sein oder sich während des Trainings verändern. Weil in dieser Arbeit Optimierer verwendet werden, die intern bei der Berechnung der Gradienten Gewichtungsfaktoren für jeden Parameter anpassen (vgl. Adam-Optimierer), wird α hier bei einem konstanten Wert belassen. Die Vorgehensweise (3.14) wird als Gradientenabstiegsverfahren (engl. *gradient descent*, *steepest descent*) bezeichnet. Im Zusammenhang mit KNN wird dieses Verfahren auch als stochastisches Gradientenabstiegsverfahren bezeichnet, da der berechnete Gradient von den gerade benutzten Trainingsdatenpunkten abhängt, welche als Zufallsgröße interpretiert werden können.

Das Gradientenabstiegsverfahren konvergiert in der Praxis oft langsam und läuft Gefahr, bei zu kleinen Gradienten zu stagnieren. Deshalb gibt es viele Ansätze, dies zu verbessern. Die im Zusammenhang mit KNN gängigen Verfahren nutzen alle nur den Gradienten und nicht die zweite Ableitung, die Hesse-Matrix (siehe auch [11]), da diese in der Fehlerrückführung vergleichsweise aufwendig zu berechnen ist. In dieser Arbeit wird der Adam-Optimierer (*adaptive moment estimation*)

als Erweiterung des Gradientenabstiegsverfahrens verwendet [61]. Da der Adam-Optimierer für jeden Parameter θ in $\boldsymbol{\theta}$ nur von dem dazugehörigen Element ϑ des Gradienten $\boldsymbol{\vartheta}$ abhängt, wird er zu Gunsten der Lesbarkeit für einen einzelnen Parameter vorgestellt. Während des Trainings wird dies für alle Parameter durchgeführt, wobei die vorgestellten Hyperparameter für alle Parameter gleich gewählt werden.

Der Adam-Optimierer aktualisiert die Parameter folgendermaßen:

$$\mu_t^{\mathrm{a}} = \beta_1\, \mu_{t-1}^{\mathrm{a}} + (1-\beta_1)\, \vartheta_t \tag{3.15}$$

$$v_t^{\mathrm{a}} = \beta_2\, v_{t-1}^{\mathrm{a}} + (1-\beta_2)\, \vartheta_t^2 \tag{3.16}$$

$$\theta_t = \theta_{t-1} - \alpha \frac{\sqrt{1-\beta_2^t}}{1-\beta_1^t} \frac{\mu_t^{\mathrm{a}}}{\sqrt{v_t^{\mathrm{a}}}}\,. \tag{3.17}$$

Dabei ist $\mu^{\mathrm{a}} \in \mathbb{R}$ der exponentiell gleitende Mittelwert des Gradienten, der mit $\mu_0^{\mathrm{a}} = 0$ initialisiert wird. Dieser führt dazu, dass vergangene Werte des Gradienten auch Einfluss auf die Parameteraktualisierung haben. Dies vermeidet eine schnelle Stagnation des Trainings in Bereichen mit kleinem Gradienten. Dieses Konzept wird auch von weiteren Optimierern, wie dem Momentum-Optimierer [96, 121] und dem Nesterov-Momentum-Optimierer [25, 86], genutzt.

Der exponentiell gleitende Mittelwert des quadratischen Gradienten $v^{\mathrm{a}} \in \mathbb{R}$, der mit $v_0^{\mathrm{a}} = 0$ initialisiert wird, führt dazu, dass sich der Gradient einzelner Parameter vergrößert, wenn diese bei den vergangenen Schritten wenig verändert wurden. Auch dies hilft bei der Überwindung flacher Bereiche der Verlustfunktion oder lokaler Minima. Das Konzept wird auch beim ADADELTA-Optimierer [140], beim ADAGRAD-Optimierer [28] und beim RMSProp-Optimierer [50] in ähnlicher Weise genutzt. Die Parameter $\beta_1 \in (0,1)$ und $\beta_2 \in (0,1)$ sind Hyperparameter und legen fest, wie stark sich die vergangenen Werte von μ^{a} und v^{a} auswirken. Der erste Bruch in (3.17) kompensiert die Initialisierung mit 0 bei der Mittelwertbildung. Um eine Division durch 0 zu vermeiden, wird in der Praxis zum Nenner des zweiten Bruchs in (3.17) eine kleine Zahl addiert (siehe [61]).

3.2.3.3 Batch-Größe

Die Optimierung erfolgt in jedem Schritt für eine bestimmte Anzahl an Datenpunkten, die als *Batch*-Größe (engl. *batch size*) bezeichnet wird. Bei KNN wird meist nicht angegeben, wie viele Trainingsschritte t durchgeführt werden, sondern, wie oft jeder Datenpunkt des Trainingsdatensatzes verwendet wird. Wird der ganze Trainingsdatensatz einmal zum Training verwendet, wird von einer Epoche gesprochen.

Werden alle Datenpunkte in jedem Trainingsschritt verwendet, hat dies den Vorteil, dass der vollständige Trainingsdatensatz an der Berechnung des Gradienten beteiligt ist und wenige Trainingsschritte benötigt werden. Ein Trainingsschritt entspricht dann einer Epoche. Von Nachteil sind die hohen Speicheranforderungen. Wird hingegen nur ein Datenpunkt je Trainingsschritt verwendet, sind die Speicheranforderungen viel geringer, jedoch entstehen hier meist Konvergenzprobleme während des Trainings, da sich die Verlustfunktion mit wechselndem Eingang ständig ändert. In der Praxis wird meist ein Kompromiss gefunden, bei dem in jedem Trainingsschritt eine Teilmenge an Datenpunkten verwendet wird [64, 72]. Sind die unterschiedlichen Datenpunkte gleichmäßig auf die Teilmengen verteilt, konvergiert auch dieses Verfahren, bei geringeren Speicheranforderungen im Vergleich zum Training mit dem gesamten Trainingsdatensatz. In dieser Arbeit wird zum Training in den meisten Fällen der gesamte Trainingsdatensatz verwendet, da die Datenpunkte eindimensionale Spektren sind, die wenig Speicher benötigen.

Ist die *Batch*-Größe größer als 1, wird, je nach Implementierung, der Mittelwert oder die Summe der Gradienten, die zu den einzelnen Datenpunkten gehören, als resultierender Gradient verwendet. Auch wenn die Richtung des resultierenden Gradienten in beiden Fällen gleich ist, so ist der Betrag im Fall einer Summe abhängig von der *Batch*-Größe. Dies kann durch die Wahl der Lernrate α kompensiert werden.

3.2.4 Batch-Normalisierung

Ein Problem beim Training mehrschichtiger neuronaler Netze ist, dass sich mit der Änderung der Parameter der vorhergehenden Schichten die Eingangswerte einer Schicht ständig ändern. Damit müssen die Parameter späterer Schichten immer wieder an die Parameter der vorherge-

henden Schichten angepasst werden. Dazu kommt, wenn nicht in jedem Schritt mit dem gesamten Trainingsdatensatz trainiert wird, eine Änderung der Eingangsgrößen aufgrund der unterschiedlichen Eingangsdatenpunkte. Um dem entgegenzuwirken, wird *Batch*-Normalisierung verwendet [55]. Die *Batch*-Normalisierung wird für jedes Neuron separat durchgeführt, weshalb es an dieser Stelle genügt, die Beziehung zwischen einem eindimensionalen Eingang $x \in \mathbb{R}$ und einem eindimensionalen Ausgang $y \in \mathbb{R}$ einer *Batch*-Normalisierung zu beschreiben. Die *Batch*-Normalisierung

$$y = w\frac{x - \hat{\mu}_{\mathcal{B}}}{\sqrt{\hat{\sigma}_{\mathcal{B}}}} + b \tag{3.18}$$

wird unmittelbar vor der Aktivierungsfunktion eingesetzt, sodass hier y dem Eingang der Aktivierungsfunktion und x dem Ausgang eines Neurons entspricht. Dabei sind $\hat{\mu}_{\mathcal{B}} \in \mathbb{R}$ und $\hat{\sigma}_{\mathcal{B}} \in \mathbb{R}$ der Stichprobenmittelwert und die Stichprobenvarianz von x. Bei der Berechnung der beiden Größen wird zwischen vollständig verbundenen Schichten (siehe 3.2.1) und Faltungsschichten (siehe 3.3.1) unterschieden. Bei vollständig verbundenen Schichten werden die Momente auf Basis aller aktuell verwendeten Datenpunkte und den daraus resultierenden Werten für x berechnet. Bei Faltungsschichten werden sie zusätzlich auf Basis aller Werte auf der gleichen Merkmalskarte (siehe 3.3) berechnet. Um eine Division durch 0 zu vermeiden, wird in der Praxis zum Nenner des Bruchs in (3.18) eine kleine Zahl addiert [55]. Damit haben die Zahlenwerte, die die darauf folgende Schicht als Eingangsgrößen bekommen, immer einen ähnlichen Wertebereich. Um dennoch die Aktivierungsfunktion optimal nutzen zu können, werden der Gewichtungsfaktor $w \in \mathbb{R}$ und der Schwellenwert $b \in \mathbb{R}$ der *Batch*-Normalisierung verwendet, die analog zu denen der Neuronen mit dem KNN trainiert werden. Für die Inferenz werden $\hat{\mu}_{\mathcal{B}}$ und $\hat{\sigma}_{\mathcal{B}}$ basierend auf dem gesamten Trainingsdatensatz auf einen festen Wert gesetzt.

Durch die *Batch*-Normalisierung kann eine höhere Lernrate gewählt werden und die Abhängigkeit von der Initialisierung der Parameter sinkt. Darüber hinaus wirkt die *Batch*-Normalisierung regularisierend [55]. Die Initialisierung von w und b sollte an die verwendete Aktivierungsfunktion angepasst sein, sodass flache Bereiche der Aktivierungsfunktion und damit kleine Gradienten vermieden werden.

3.3 Faltungsnetze

Es gibt KNN, die nicht nur aus vollständig verbundenen Schichten bestehen, bei denen die Matrix **W** der Gewichte vollständig besetzt ist. Stattdessen nutzen sie die diskrete Faltung, weshalb diese KNN als Faltungsnetze (engl. *convolutional neural networks*, CNN) bezeichnet werden [68]. Die Faltungsoperation lässt sich als dünn besetzte Gewichtematrix darstellen [91, 92]. Dadurch besitzt eine Faltungsschicht, bei gleicher Ein- und gleicher Ausgangsgröße, wesentlich weniger Parameter als eine vollständig verbundene Schicht. Außerdem werden die gleichen Gewichte an unterschiedlichen Stellen verwendet, was auch als lokale Verbindungen bezeichnet wird [70]. Dadurch bleibt die Information erhalten, an welcher Stelle ein Merkmal auftritt (z. B. an welchem Ort bei Bildern oder bei welcher Frequenz bei den Spektren aus Kapitel 2).

Vor allem in der Bildverarbeitung konnten große Erfolge mit CNN erzielt werden. Wichtige Aufgabenstellungen sind dabei die Klassifikation, die Detektion und die Segmentierung. Bekannte Netzarchitekturen aus dem Bereich der Bildverarbeitung sind das AlexNet [68], das VGG-Netz [116] und das ResNet [46]. Neben der Faltungsschicht spielt auch die *Pooling*-Schicht eine wichtige Rolle bei CNN. Beide werden im Folgenden genauer beschrieben.

3.3.1 Faltungsschicht

Da in dieser Arbeit vor allem eindimensionale Spektren (siehe Kapitel 2) als Daten verarbeitet werden, wird hier die Faltungsschicht für eindimensionale Daten vorgestellt. Diese lässt sich jedoch einfach auf mehrdimensionale Daten erweitern. Dazu muss ein mehrdimensionaler Faltungskern eingesetzt und die Faltung entlang aller relevanten Dimensionen durchgeführt werden. Bei einer Faltungsschicht wird Gleichung (3.6) zu

$$\mathbf{y} = \boldsymbol{\varphi}\left(\mathbf{x} * \mathbf{w} + b \cdot \mathbf{1}\right) \ . \tag{3.19}$$

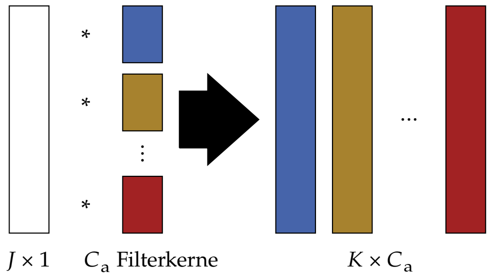


Abbildung 3.4 Die Faltung einer Merkmalskarte mit C_a Filterkernen resultiert in genau so vielen Merkmalskarten am Ausgang [V7].

Dabei sind $\mathbf{w} \in \mathbb{R}^I$ der Faltungskern, $\mathbf{1} \in \mathbb{R}^K$ ein Vektor, bei dem alle Elemente eine 1 sind, und $*$ der Faltungsoperator. Durch

$$y_k = \varphi\left(\sum_{i=1}^{I} x_{\tilde{j}} \cdot w_i + b\right) \quad \text{mit: } \tilde{j} = k - \frac{1}{2}(I-1) + i \tag{3.20}$$

wird dieser definiert, wie er in der Praxis in CNN genutzt wird. Dabei wurde zu Gunsten der Lesbarkeit die Spiegelung des Faltungskerns weggelassen. Dies wird bei CNN häufig auch in der Praxis so gemacht, da die Gewichte ohnehin während des Trainings gelernt werden und die Spiegelung damit keinen Einfluss hat. Des Weiteren werden die Faltungsschichten hier so verwendet, dass die Ein- und Ausgangsdatenpunkte die gleiche Größe haben. Dazu wird für x eine 0 angenommen, wenn $\tilde{j}$ außerhalb der Eingangswerte liegt (engl. *zero padding*):

$$x_{\tilde{j}} = 0 \quad \text{für: } \tilde{j} < 1, \tilde{j} > J\,. \tag{3.21}$$

Der Ausgang einer Faltungsschicht wird als Merkmalskarte (engl. *feature map*) bezeichnet. In der Praxis genügt ein einzelner Filterkern oft nicht. Stattdessen wird der Eingang der Schicht mit $C_a \in \mathbb{R}$ Filterkernen gefaltet. Jede Faltung entspricht einer Merkmalskarte am Ausgang der Schicht (siehe Abbildung 3.4). In tieferen Schichten sind dadurch mehrere Merkmalskarten am Eingang einer Schicht vorhanden. Bei Farbbildern sind auch am Eingang des CNNs mehrere Merkmalskarten vorhanden, da die Farbkanäle als solche behandelt werden. Bei den hier untersuchten Spektren hingegen gibt es nur eine Merkmalskarte am Eingang des

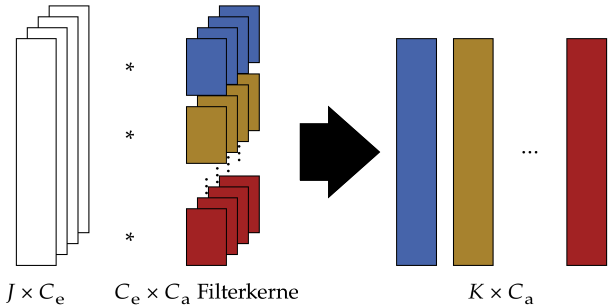


Abbildung 3.5 Sind C_e Merkmalskarten am Eingang vorhanden, wird je Merkmalskarte an Ein- und Ausgang ein Filterkern benutzt. Die dazugehörige Merkmalskarte am Ausgang ist die Summe der gefalteten Merkmalskarten am Eingang [V7].

CNNs. Jede der $C_\mathrm{e} \in \mathbb{R}$ Merkmalskarten am Eingang einer Schicht wird pro Merkmalskarte am Ausgang mit je einem Filterkern gefaltet. Für die entsprechende Merkmalskarte am Ausgang werden die Resultate addiert (siehe Abbildung 3.5). Für jede Merkmalskarte am Ausgang einer Schicht gibt es genau einen Schwellenwert b, der auf alle Werte der Merkmalskarte addiert wird.

Bei vielen CNN wechseln sich die Faltungsschichten mit *Pooling*-Schichten ab, die im nächsten Abschnitt vorgestellt werden.

3.3.2 Pooling-Schicht

Beim *Pooling* wird eine lokale Nachbarschaft einer Merkmalskarte in einem einzigen Wert zusammengefasst. Dabei wird das *Pooling* entlang der gleichen Dimensionen wie die Faltung durchgeführt. Die Merkmalskarten werden dabei separat behandelt. Die Größe der Nachbarschaft kann frei bestimmt werden, wobei üblicherweise ein rechteckiges Fenster gewählt wird.

Es gibt unterschiedliche Möglichkeiten, die Zahlen zusammenzufassen, wie z. B. durch Bildung des Mittelwerts oder des Maximums. Letzteres hat sich dabei besonders bewährt und wird als *Max-Pooling* [112] bezeichnet (Beispiel siehe Abbildung 3.6).

Durch *Pooling* werden die Merkmalskarten verkleinert, was mehrere Vorteile mit sich bringt. Zunächst sinkt dadurch der Rechenaufwand,

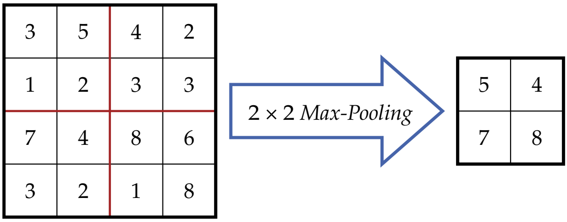


Abbildung 3.6 Zahlenbeispiel für *Max-Pooling* mit Fenstergröße 2×2 einer zweidimensionalen Merkmalskarte.

weil in den folgenden Schichten mit kleineren Merkmalskarten gearbeitet wird, was die Anzahl an zu berechnenden Operationen reduziert. Der Speicherbedarf verringert sich ebenfalls, da die zu speichernden Merkmalskarten kleiner sind. Darüber hinaus wird das Netz unempfindlicher gegen kleine Abweichungen [71], die z. B. durch Rauschen verursacht werden können, wodurch eine bessere Generalisierung erzielt werden kann (siehe Abschnitt 3.1.5). Zuletzt wird das lokale Sichtfeld, also der Bereich, den ein Faltungskern abdecken kann, in den hinteren Schichten größer, da die Information eines größeren Bereichs auf einen kleineren abgebildet wird. Letzteres erlaubt die Verwendung kleinerer Filterkerne, was sich wiederum positiv auf den Ressourcenbedarf auswirkt und ebenfalls Überanpassung entgegenwirkt.

3.4 Generative Adversarial Networks

Generative Adversarial Networks sind künstliche neuronale Netze, die unüberwacht trainiert werden und die Verteilung des Trainingsdatensatzes lernen [36]. Damit ist es möglich, weitere Datenpunkte zu erzeugen, die der gleichen Verteilung genügen. *Generative Adversarial Networks* bestehen aus zwei Teilnetzen, einem Generator $\mathcal{G}$ und einem Diskriminator $\mathcal{D}$. Die Eingangsdaten des Generators sind zufällige Werte, die im Zufallsvektor $\mathbf{z}$ zusammengefasst sind. Die Eingangsdaten des Diskriminators sind entweder die Ausgangsdaten des Generators oder Datenpunkte $\mathbf{x}$ aus dem Trainingsdatensatz. Der Ausgang des Diskriminators ist ein Skalar, welches angibt, ob es sich beim Eingangsdatenpunkt um einen generierten oder einen realen handelt. Der Aufbau eines GANs wird in Abbil-

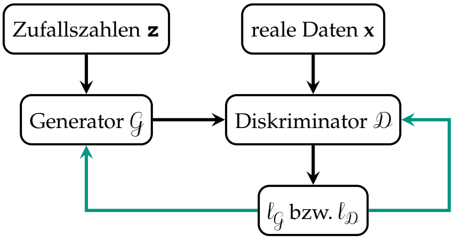


Abbildung 3.7 Aufbau eines GANs [V10]. Schwarze Pfeile stehen für den Datenfluss in Vorwärtsrichtung, grüne für die Fehlerrückführung.

dung 3.7 dargestellt. Das Ziel des Generators ist es, Daten zu generieren, die den Trainingsdaten möglichst ähnlich sind, sodass der Diskriminator nicht zwischen echten und generierten Daten unterscheiden kann. Der Diskriminator versucht hingegen, in seiner Aufgabe immer besser zu werden.

Die Parameter der beiden Teilnetze werden abwechselnd trainiert. Die Verlustfunktion beim Training des Generators lautet

$$l_\mathcal{G} = -\sum_{n=1}^{N} \log\left(\mathcal{D}\left(\mathcal{G}\left(\mathbf{z}_n\right)\right)\right) \tag{3.22}$$

und die beim Training des Diskriminators

$$l_\mathcal{D} = -\sum_{n=1}^{N} \left[\log\left(\mathcal{D}\left(\mathbf{x}_n\right)\right) + \log\left(1 - \mathcal{D}\left(\mathcal{G}\left(\mathbf{z}_n\right)\right)\right)\right], \tag{3.23}$$

wenn mit jeweils N Datenpunkten trainiert wird. Dabei ist (3.23) so zu verstehen, dass der erste Summand für alle realen und der zweite für alle generierten Datenpunkte verwendet wird und der jeweils andere verschwindet. Der beim Training vorgegebene wahre Ausgangswert 1 steht beim Diskriminator dafür, dass ein Datenpunkt für echt befunden wird. Beim Wert 0 hält der Diskriminator den Datenpunkt für generiert. Der tatsächliche Ausgangswert des Diskriminators ist irgendwo dazwischen. Die Klassifikation findet mit einer Grenze, meist bei 0,5, statt. Durch abwechselndes Training verbessern sich Generator und Diskriminator, bis im Idealfall die Daten des Generators so authentisch sind, dass der

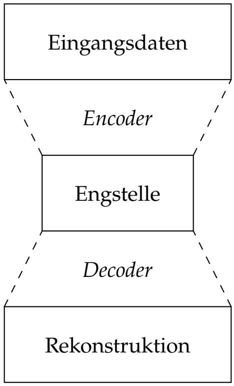


Abbildung 3.8 Schematische Darstellung eines *Autoencoders* [V2].

Diskriminator nicht mehr in der Lage ist, zwischen echten und generierten Daten zu unterscheiden, obwohl er diese Aufgabe hervorragend beherrscht.

3.5 Autoencoder

Bei *Autoencodern* [1, 35, 65, 76] handelt es sich ebenfalls um unüberwacht trainierte KNN (siehe Abschnitt 3.1). Beim Training werden identische Ein- und Ausgangsdaten verwendet. Meist wird ein *Autoencoder* mit einer Engstelle realisiert, die eine niedrigere Dimension als die Eingangsdaten hat. Der Teil des KNNs vor der Engstelle wird als *Encoder* bezeichnet, der Teil, der die Daten rekonstruiert, als *Decoder* (siehe Abbildung 3.8). Ziel ist es, eine Kompression mit möglichst wenig Informationsverlust zu erreichen. Dazu wird beim Training der Fehler zwischen den Eingangsdaten und den im *Decoder* aus der komprimierten Darstellung rekonstruierten Daten minimiert.

Eine wichtige Klasse von *Autoencodern* ist der *Variational Autoencoder* [62]. Dabei handelt es sich um einen *Autoencoder*, dessen *Encoder* die Momente einer Wahrscheinlichkeitsverteilung als Ausgangsgrößen hat (siehe Abbildung 3.9). In der Praxis wird hier meist eine mehrdimensionale Normalverteilung verwendet, deren Elemente stochastisch

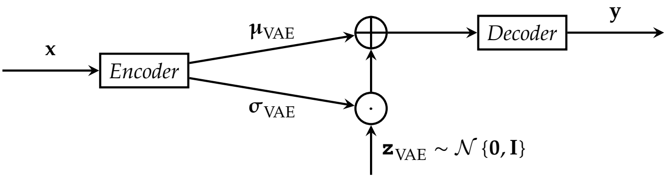


Abbildung 3.9 Schematische Darstellung eines *Variational Autoencoders*. Dabei wird der Zufallsvektor $\mathbf{z}_{VAE} \in \mathbb{R}^Z$ für die Realisierungen der Eingangsdaten des *Decoders* genutzt.

unabhängig sind. Damit sind die Ausgänge des *Encoders* der Mittelwertvektor $\boldsymbol{\mu}_{VAE} \in \mathbb{R}^Z$ und die Hauptdiagonale der Kovarianzmatrix als Vektor $\boldsymbol{\sigma}_{VAE} \in \mathbb{R}^Z$. Der *Decoder* erhält Realisierungen der entsprechenden Zufallsvariablen als Eingangsgrößen (siehe Abbildung 3.9). Solche KNN werden benutzt, um zusätzliche Daten zu erzeugen, die ähnliche Eigenschaften wie die Trainingsdaten besitzen. Beim Training des *Variational Autoencoders* wird nicht nur der Fehler der Rekonstruktion minimiert. Es wird auch eine Wahrscheinlichkeitsverteilung vorgegeben, die im Laufe des Trainings von den Ausgangsgrößen des *Encoders* angenommen werden soll. Dazu wird die Verlustfunktion entsprechend angepasst. In der Praxis wird meist die mehrdimensionale Standardnormalverteilung genommen und die Erweiterung der Kostenfunktion mit der Kullback-Leibler-Divergenz realisiert [69].

Eine Erweiterung dazu wäre der *conditional Variational Autoencoder* [24], bei dem zusätzlich Bedingungen an den *Decoder* übergeben werden, um die Möglichkeiten der durch den *Decoder* erzeugten Daten einzugrenzen bzw. vorzugeben.

4 Verwandte Arbeiten

In diesem Kapitel werden bestehende Verfahren vorgestellt, die eine spektrale Entmischung mit Hilfe von KNN durchführen. Dabei wird in Abschnitt 4.1 zunächst auf unüberwacht trainierte Verfahren eingegangen, worüber die meisten Arbeiten existieren. Im Anschluss werden Verfahren, die überwacht trainiert werden, in Abschnitt 4.2 dargestellt. Im Abschnitt 4.3 werden Verfahren zur Modellierung der Spektrenvariabilität und zur Augmentierung spektraler Datensätze vorgestellt.

4.1 Unüberwachtes Training

Damit die unüberwachten Verfahren, bei denen nur Spektren und keine zugehörigen Anteilsvektoren vorliegen, funktionieren, muss ein Mischmodell genutzt werden, da der Zusammenhang zwischen Mischspektren und Anteilsvektoren nicht aus den Daten gelernt werden kann. Die meisten Verfahren nutzen das LMM, wobei auch Ansätze existieren, die nichtlineare Mischmodelle verwenden. Der Großteil der unüberwachten Verfahren ist nach dem Prinzip eines *Autoencoders* (siehe Abschnitt 3.5) aufgebaut.

4.1.1 Autoencoder

Die Verfahren, die nach dem Prinzip eines *Autoencoders* aufgebaut sind, nutzen als *Encoder* ein beliebiges KNN, das als Eingang ein Spektrum und als Ausgang einen Anteilsvektor hat. Der *Decoder* hat dementsprechend den Anteilsvektor als Eingang und ein rekonstruiertes Spektrum als Ausgang. An der Engstelle am Ausgang des *Encoders* wird auf diese Weise eine niedrigdimensionale Darstellung des Spektrums erzwungen (siehe Abschnitt 3.5), die im Allgemeinen nicht dem Anteilsvektor entspricht. Damit an der Engstelle gerade die Anteilsvektoren ermittelt

werden, wird der *Decoder* so ausgelegt, dass er ein Mischmodell realisiert. Dadurch ist es mit ihm nicht möglich, eine beliebige Funktion abzubilden [89]. Stattdessen beinhalten seine Netzparameter die Parameter des verwendeten Mischmodells und insbesondere die Reinspektren. Die Sicherstellung der Nebenbedingungen (2.5) und (2.6) erfolgt durch entsprechende Auslegung des *Encoders*, wie bspw. die Verwendung einer *Softmax*-Schicht (5.2).

Wird das LMM verwendet, lässt sich der *Decoder* mit einer einzelnen vollständig verbundenen Schicht gemäß (3.6) realisieren, die dem LMM entspricht. Diese vollständig verbundene Schicht muss sich direkt hinter der Schicht, welche die Anteile als Ausgänge hat, befinden und die Anzahl der Neuronen muss der Anzahl der Wellenlängenkanäle entsprechen. Des Weiteren darf nach der Schicht keine nichtlineare Aktivierungsfunktion verwendet werden und der Schwellenwert $\mathbf{b}$ muss den Wert $\mathbf{0}$ haben. Wird der *Autoencoder* schließlich so trainiert, dass der Fehler zwischen dem Spektrum am Eingang und dem rekonstruierten Spektrum am Ausgang minimiert wird, dann entspricht die Gewichtematrix $\mathbf{W}$ der Schicht gerade der geschätzten Reinspektrenmatrix $\hat{\mathbf{M}}$ des LMMs. An der Engstelle des *Autoencoders* wird gleichzeitig der Anteilsvektor $\hat{\mathbf{a}}$ des am Eingang anliegenden Spektrums geschätzt. Da mit Hilfe eines ganzen Datensatzes trainiert wird, wird die Gewichtematrix anhand aller Spektren im Datensatz ermittelt.

Guo et al. [40] nutzen diesen Ansatz mit je einer vollständig verbundenen Schicht im *Encoder* und im *Decoder*. Dazu kommt eine zusätzliche Schicht zur Verringerung des Rauschens am Eingang. Das KNN wird mit verrauschten Spektren am Eingang trainiert, um so eine Entrauschung zusätzlich zur spektralen Entmischung zu lernen. Palsson et al. [90] erweitern diesen Ansatz (ohne Reduzierung des Rauschens), indem mehr vollständig verbundene Schichten im *Encoder* verwendet werden. Damit sind mehr Freiheitsgrade gegeben, um aus den Spektren die Anteilsvektoren zu berechnen. Die Rekonstruktion findet auch hier mit der oben beschriebenen vollständig verbundenen Schicht statt. Vijayashekhar et al. [133] nutzen ebenfalls einen *Autoencoder*, der aus einer vollständig verbundenen Schicht mit LMM im *Decoder* für die spektrale Entmischung besteht. Er wird durch einen zweiten *Autoencoder* in Reihe ergänzt, um Rauschen zu reduzieren. Die Idee dabei ist, dass der Ausgang des ersten

Autoencoders die vom Rauschen bereinigte Version des Eingangsspektrums ist und der zweite dieses wieder für die Rekonstruktion hinzufügt.

Auch Su et al. [119] nutzen den Ansatz, der das LMM durch den *Decoder* realisiert. Als *Encoder* wird jedoch eine Kette an *Autoencodern* genutzt, die sich besonders gut für verrauschte Spektren eignet. Dies wird in einer weiteren Arbeit erweitert, indem am Ende ein *Variational Autoencoder* verwendet wird, der Vorteile für die Einhaltung der Nebenbedingungen bei der spektralen Entmischung mit sich bringt [120]. Hua et al. [53] nutzen ebenfalls den beschriebenen *Decoder*, wobei im *Encoder* nicht nur einzelne Spektren, sondern auch Spektren benachbarter Pixel im Hyperspektralbild genutzt werden. Das KNN besteht aus zwei Strängen, von denen einer Einzelspektren und der andere Spektren inklusive Nachbarschaft mit einer zweidimensionalen Faltung verarbeitet. In einer späteren Schicht werden die Ausgänge vereint. Nach dem gleichen Prinzip im *Decoder*, aber mit einem *Transformer Network* als *Encoder*, wird die spektrale Entmischung von Ghosh et al. [33] durchgeführt. Fang et al. [31] nutzen ein CNN, wobei auch hier am Ausgang eine vollständig verbundene Schicht für das LMM genutzt wird. Dabei wird mit Bayes'schen KNN gearbeitet, um Rauschen besser modellieren zu können.

Darüber hinaus kann der *Autoencoder*-Ansatz mit Hilfe eines GANs regularisiert werden, was als *Adversarial Autoencoder* bezeichnet wird [75]. Holland und Du [52] regularisieren damit den *Decoder*, also die Rekonstruktion der Spektren. Dabei wird ein zweiteiliges Training genutzt. Im ersten Teil werden die Anteilsvektoren aus den Spektren berechnet, während die Gewichte des Diskriminators unveränderlich bleiben. Im zweiten Teil wird der Diskriminator berücksichtigt und die Anteilsvektoren werden aus zufälligen Zahlen erzeugt. Jin et al. [58] hingegen nutzen den Diskriminator, um die errechneten Anteilsvektoren zu regularisieren, was vor allem im Kontext der Fernerkundung sinnvoll ist.

Auch wenn das LMM für viele Anwendungen gute Ergebnisse liefert, ist die Verwendung nichtlinearer Mischmodelle in vielen Fällen erforderlich (siehe Abschnitt 2.4). Auch dafür existieren Verfahren basierend auf dem *Autoencoder*-Ansatz.

Für die Unterstützung nichtlinearer Zusammenhänge nutzen Wang et al. [135] zusätzliche vollständig verbundene Schichten im *Decoder* nach der Schicht, die das LMM realisiert. Als Reinspektren werden weiterhin

die Gewichte der vollständig verbundenen Schicht ohne Aktivierungsfunktion genutzt. Allerdings kann deren Ausgang mit den zusätzlichen Schichten weiterverarbeitet werden, sodass zwischen Anteilsvektor und rekonstruiertem Spektrum kein linearer Zusammenhang mehr bestehen muss. Zhao et al. nutzen einen ähnlichen Ansatz mit dem Unterschied, dass der Ausgang der vollständig verbundenen Schicht, die das LMM realisiert, auf den Ausgang des Netzes addiert wird. Damit ist das rekonstruierte Spektrum die Summe einer linearen und einer nichtlinearen Komponente bezüglich der Anteile. Darüber hinaus werden im *Encoder* sogenannte *Long-Short-Term-Memory*-Schichten [143] bzw. Faltungsschichten genutzt [144].

Rasti et al. [102] nutzen den *Autoencoder*-Ansatz mit einer vollständig verbundenen Schicht im *Decoder*. In der Verlustfunktion werden (2.10) und (2.11) genutzt, um die Reinspektren in die Albedo-Domäne zu transformieren und damit eine spektrale Entmischung nach LMM durchzuführen.

4.1.2 Weitere Verfahren mit linearem Mischmodell

Es existieren Ansätze, die auf dem LMM basieren, aber keinen *Autoencoder* nutzen. Bei Xiong et al. [138] wird das Optimierungsproblem (2.16) des LMMs umformuliert, sodass es für alle Datenpunkte eines Datensatzes gleichzeitig optimiert wird. Dabei ist zu beachten, dass auch die Reinspektrenmatrix unbekannt ist. Dieses Problem lässt sich mit der nichtnegativen Matrixfaktorisierung lösen, wobei alternierend die Reinspektren und die Anteilsvektoren aller Datenpunkte optimiert werden. Xiong et al. [138] bilden dies mit einem KNN nach, wobei jede Schicht des KNNs einer Iteration entspricht. Dabei werden Parameter definiert, die trainiert werden. Auch Qian et al. [99] nutzen ein KNN, um ein iteratives Verfahren basierend auf dem LMM nachzubilden. Dabei entsprechen die einzelnen Schichten je einer Iteration. Rasti et al. nutzen im LMM ein KNN, das die Anteilsvektoren des ganzen Datensatzes als Funktion einer festen zufälligen Eingangsgröße modelliert [101, 103, 104].

Es wird deutlich, dass die unüberwacht trainierten Verfahren bereits ausführlich untersucht worden sind. Alle unüberwacht trainierten Verfahren unterscheiden sich von den Verfahren, die in dieser Arbeit vorgestellt werden, dahingehend, dass sie die Reinspektren ebenfalls aus

den Daten ermitteln. Damit führen sie auch eine unüberwachte spektrale Entmischung durch. Da es sich bei den hier vorgestellten Verfahren um überwachte Verfahren handelt und die Reinspektren ohnehin im Trainingsdatensatz vorhanden sind, müssen diese nicht ermittelt werden. Überwachtes Training hat den Vorteil, dass die Mischzusammenhänge direkt aus den Daten gelernt werden und damit beliebig sein können.

In Kapitel 6 dieser Arbeit werden Verfahren vorgestellt, die mit Hilfe von Mischmodellen Datensätze erzeugen, die für das überwachte Training eines CNNs für die spektrale Entmischung genutzt werden. Damit wird indirekt, wie bei den unüberwachten Verfahren, ein Mischmodell für die spektrale Entmischung mit KNN genutzt. Es wird jedoch nicht in die Netzstruktur integriert. Damit können nichtlineare Mischmodelle einfacher umgesetzt werden.

Auch überwachtes Training wird bereits, wie im nächsten Abschnitt vorgestellt, für die spektrale Entmischung genutzt, wobei nicht jeder Ansatz ohne Mischmodell auskommt.

4.2 Überwachtes Training

Plaza et al. [94] nutzten bereits 2004 KNN für die spektrale Entmischung. Sie teilen das Problem der spektralen Entmischung in einen linearen und einen nichtlinearen Teil auf. Der nichtlineare Teil wird dabei mit Hilfe eines *Multilayer Perceptrons* (siehe Abschnitt 3.2.1) aus Daten gelernt. Später verwenden Plaza et al. [95] ein *Multilayer Perceptron*, welches die spektrale Entmischung datenbasiert durchführt. Dabei beschäftigen sie sich damit, welche Pixel in einem hyperspektralen Bild besonders gut als Trainingsdaten geeignet sind.

Xu et al. [139] setzen KNN zur spektralen Entmischung ein. Dabei werden die Trainingsdaten mit Mischmodellen ähnlich wie in Kapitel 6 erzeugt, jedoch wird die Spektrenvariabilität (anders als in der vorliegenden Arbeit) nicht berücksichtigt.

Wan et al. [134] nutzen ein KNN, welches aus Faltungsschichten und vollständig verbundenen Schichten besteht und damit der eindimensionalen Version des Netzes in Kapitel 5 ähnlich ist. Das lässt vermuten, dass die hier verwendete Netzstruktur für die spektrale Entmischung auch für weitere Datensätze sinnvoll ist.

Darüber hinaus wird bei hyperspektralen Bildern oft eine dreidimensionale Faltung eingesetzt [124, 125, 142], wie sie beim CNN in Kapitel 5 ebenfalls verwendet werden kann. Damit wird die Nutzung der Information der örtlichen Dimensionen ermöglicht. Dies ist vor allem in der Fernerkundung relevant und wird in dieser Arbeit nicht näher untersucht, da die verwendeten Datensätze kaum Informationen entlang der örtlichen Dimensionen beinhalten, weil es sich um Pulvermischungen handelt. Tulczyjew et al. [131] nutzen dazu eine spezielle Netzarchitektur, bei der es je einen Pfad mit ein-, zwei- und dreidimensionalen Faltungen gibt, die am Ende fusioniert werden.

Die meisten bestehenden Verfahren beschäftigen sich damit, ein geeignetes KNN und ein geeignetes Trainingsverfahren für die spektrale Entmischung zu finden. In dieser Arbeit besteht der Schwerpunkt dagegen darin, die Trainingsdaten zu erzeugen bzw. zu erweitern, sodass die KNN leichter bzw. besser trainiert werden können. Dabei wird insbesondere immer die Spektrenvariabilität miteinbezogen.

4.3 Berücksichtigung der Spektrenvariabilität

Die bisher vorgestellten Ansätze modellieren die Spektrenvariabilität nicht direkt. Bei den Verfahren, die ohne Mischmodell und nur mit Hilfe von Daten trainiert werden, wird sie trotzdem berücksichtigt, wenn sie durch die Trainingsdaten abgedeckt wird. Bei den bisher vorgestellten unüberwachten Verfahren wird für jeden Reinstoff ein Reinspektrum gelernt. Damit kommt bei gleichem geschätzten Anteilsvektor auch immer die gleiche Rekonstruktion heraus. Die Spektrenvariabilität kann dadurch berücksichtigt werden, dass in dem Teil, der die Anteilsvektoren schätzt (meist der *Encoder*), für unterschiedliche Spektren der gleiche Anteilsvektor ermittelt wird. Im Folgenden werden Verfahren vorgestellt, die zusätzliche Möglichkeiten nutzen, um die Spektrenvariabilität zu modellieren.

Bei Zhang et al. [141] wird zusätzlich zu den Reinspektren eine Bibliothek an spektralen Verläufen je Reinstoff gelernt, welche die Spektrenvariabilität charakterisieren. Dabei wird das lineare Mischmodell mit einer gewichteten Summe dieser Verläufe erweitert. Die Gewichte werden wie

die Anteile geschätzt. Die Optimierung erfolgt dabei für den ganzen Datensatz bzw. für ein ganzes Hyperspektralbild.

Shi et al. [114] nutzen einen *Autoencoder*, dessen *Encoder* neben dem Anteilsvektor auch die Momente eines Zufallsvektors als Ausgangsgrößen besitzt. Damit werden im *Decoder* Reinspektren zufällig erzeugt, womit die Spektrenvariabilität berücksichtigt wird. Mit dem LMM wird aus den Reinspektren und dem Anteilsvektor die Rekonstruktion berechnet, über die der *Autoencoder* trainiert wird. Eine Erweiterung dessen teilt den *Decoder* in einen Block je Reinstoff, von denen jeder einen Diskriminator für die Reinspektren [115] besitzt. Durch diese GAN-Struktur sollen die Reinspektren möglichst authentisch aus den Daten gelernt werden. Außerdem werden hier nicht einzelne Pixel, sondern Bereiche um die Pixel miteinbezogen.

Borsoi et al. [7] nutzen einen *Variational Autoencoder*, um die Reinspektrenvariabilität zu modellieren. Dazu nutzen sie vorhandene Reinspektrenmengen. Diese Modellierung wird dann genutzt, um neue Reinspektren zu erzeugen und die vorhandenen Reinspektrenmengen zu erweitern [8, 9].

Die Verfahren, die in Kapitel 7 erarbeitet werden, erweitern ebenfalls Datensätze unter Berücksichtigung der Spektrenvariabilität. Im Gegensatz zu bestehenden Verfahren erfolgt die Erweiterung nicht nur für die Reinspektren, sondern auch für Mischspektren. Dabei können Anteilsvektoren vorgegeben werden, für die neue Spektren erzeugt werden, wobei eine stochastische Komponente die Spektrenvariabilität berücksichtigt. Dies ist notwendig, da die Erzeugung von realen Stoffgemischen für einen Trainingsdatensatz meist mit viel Aufwand verbunden ist. Auch bei der modellbasierten Datenerzeugung in Kapitel 6 wird die Spektrenvariabilität durch eine stochastische Komponente realisiert.

5 Faltungsnetz für die spektrale Entmischung

In diesem Kapitel wird ein CNN vorgestellt, das im weiteren Verlauf der Arbeit zur Bewertung der Verfahren aus den Kapiteln 6 und 7, welche die Erzeugung bzw. Erweiterung der Trainingsdatensätze beinhalten, verwendet wird. Dieses Netz ist zunächst für die Detektion von Reinstoffen in Mischungen [V3] und später für die spektrale Entmischung [V4] vorgestellt worden. Darüber hinaus wird es in späteren Veröffentlichungen zur Bewertung der Augmentierungs- und Datenerzeugungsverfahren genutzt [V6–V10]. Das vorgestellte CNN bringt Eigenschaften mit sich, die es für die spektrale Entmischung geeignet machen. Es hat sich in Kombination mit den in Kapitel 8 verwendeten Datensätzen bewährt, aber nicht den Anspruch, für alle spektralen Datensätze gute Ergebnisse bei der spektralen Entmischung zu liefern. Allerdings kann für weitere Datensätze, basierend auf den hier vorgestellten Prinzipien, ein geeignetes Netz ermittelt werden.

Das CNN ist in der Lage, Hyperspektralbilder direkt als dreidimensionale Datenwürfel zu verarbeiten, kann jedoch auch für einzelne Spektren verwendet werden. In dieser Arbeit wird es für einzelne Spektren verwendet, da die untersuchten Datensätze wenig Information entlang der räumlichen Dimensionen enthalten [V4]. Darüber hinaus werden bei der Erzeugung und Augmentierung der Datensätze nur die spektralen Eigenschaften herangezogen. Da die Spektrenvariabilität über zufällige Komponenten modelliert wird, ist eine Zuordnung einzelner Spektren zu einem bestimmten Ort auch nicht möglich. Im Allgemeinen kann es bei Hyperspektralbildern jedoch durchaus lohnend sein, diese Informationen miteinzubeziehen [3, 124, 125, 131, 142].

In Abschnitt 5.1 wird das Netz für dreidimensionale Eingangsdaten vorgestellt. Für die Verwendung mit Spektren als Eingangsdaten müssen nur Parameter entsprechend gewählt werden, worauf in Abschnitt 5.3

eingegangen wird. Für die spektrale Entmischung spielt das Einhalten der Nebenbedingungen (2.5) und (2.6) eine wichtige Rolle. Wie diese über die Wahl der Ausgangsschichten erreicht werden kann, wird in Abschnitt 5.2 beschrieben.

5.1 Netzarchitektur

Anders als bei der Klassifikation von Bildern, wo die etablierten CNN, wie z. B. das AlexNet [68], das VGG-Netz [116] und das ResNet [46], große Erfolge erzielt haben, gibt es im Bereich der spektralen Entmischung oft nur wenig Trainingsdaten. Diese reichen nicht aus, um die große Anzahl der benötigten Parameter jener Netze ohne Überanpassung zu trainieren. Zusätzlich sind die Datenpunkte Spektren und damit im Vergleich zu Farbbildern kleiner. Darüber hinaus fallen die Merkmale eines eindimensionalen Spektrums weniger komplex aus. Aus diesen Gründen ist die Verwendung eines kleineren CNNs notwendig [81, 127].

Das vorgestellte CNN besteht, ähnlich wie AlexNet [68] und das VGG-Netz [116], aus zwei Teilen: Im ersten Teil befinden sich Faltungsschichten zum Herausarbeiten der Merkmale, während sich im zweiten Teil vollständig verbundene Schichten befinden, die die Merkmale auf die Ausgangsgrößen, in dieser Arbeit die relativen Anteile, abbilden.

Das CNN hat als Eingangsdaten hyperspektrale Datenwürfel. Die Ausgangsdaten beinhalten jeweils P Intensitätsbilder, eines für jeden Reinstoff. Diese haben die gleiche örtliche Auflösung wie die Eingangsdaten und geben jeweils für jedes Pixel die relativen Anteile der enthaltenen Reinstoffe an.

Der erste Teil besteht aus Faltungsschichten (siehe Abschnitt 3.3.1), die eine dreidimensionale Faltung durchführen (siehe Abbildung 5.1). Dadurch können Merkmale extrahiert werden, die nicht nur einzelne Spektren betreffen, sondern auch deren Nachbarschaft. Es können zusätzlich örtliche Merkmale extrahiert werden. Abhängig von der Größe der Merkmale ist die Größe der Faltungskerne zu wählen. Nach der Faltung folgt eine *Batch*-Normalisierung (siehe 3.2.4) und die logistische Aktivierungsfunktion (siehe 3.2.2). Letztere hat in den Experimenten bei der spektralen Entmischung bessere Ergebnisse geliefert als die ReLU-Funktion. Dies führt zwar zu einer längeren Trainingsdauer, jedoch fällt

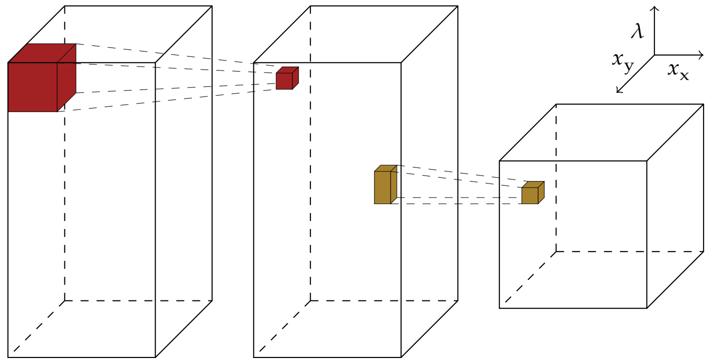


Abbildung 5.1 Ein Block des ersten Teils des CNNs: Eine dreidimensionale Faltung (rot) kombiniert mit *Max-Pooling* entlang der spektralen Dimension (braun) [V4]. Für eine übersichtlichere Darstellung werden nur eine Merkmalskarte und ein Faltungskern gezeigt.

diese bei den hier verwendeten kleinen Datenmengen dennoch nicht sehr lang aus. Das liegt vermutlich daran, dass die Anteilsvektoren als Ausgangsgrößen kontinuierliche Werte annehmen können. Während die logistische Aktivierungsfunktion stetig differenzierbar ist, besitzt die ReLU-Funktion einen Knick (siehe Abbildung 3.3).

Nach der Aktivierungsfunktion folgt eine *Max-Pooling*-Schicht (siehe Abschnitt 3.3.2). Das *Max-Pooling* erfolgt nur entlang der spektralen Richtung. Dadurch bleibt die örtliche Auflösung der hyperspektralen Bilder erhalten, was eine ortsaufgelöste Angabe der relativen Materialanteile am Ausgang möglich macht. Nach der *Max-Pooling*-Schicht folgt die nächste Faltungsschicht und ein neuer Block des ersten Teils beginnt. Am Ausgang des ersten Teils liegen mehrere dreidimensionale Merkmalskarten vor.

Die dreidimensionalen Merkmalskarten werden am Eingang des zweiten Teils entlang der spektralen Dimension in einzelne zweidimensionale Merkmalskarten aufgeteilt. Daraus resultieren viele zweidimensionale Merkmalskarten, deren Anzahl dem Produkt aus der Anzahl der dreidimensionalen Merkmalskarten und deren Länge in spektraler Dimension entspricht (siehe Abbildung 5.2). Auf diese zweidimensionalen Merkmalskarten wird eine 1×1-Faltung angewendet. Dabei findet aufgrund der Größe des Faltungskerns keine eigentliche Faltung statt, jedoch wer-

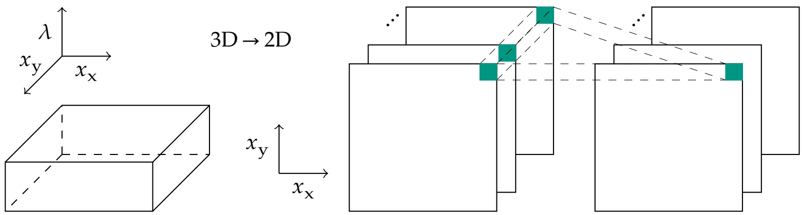


Abbildung 5.2 Datentransformation zwischen erstem und zweitem Teil des neuronalen Netzes (3D → 2D) gefolgt von einer Schicht des zweiten Teils: Eine vollständig verbundene Schicht entlang der spektralen Dimension (grün) [V4]. Für eine übersichtlichere Darstellung wird im Dreidimensionalen nur eine Merkmalskarte und im Zweidimensionalen ein Faltungskern gezeigt.

den weiterhin die Merkmalskarten, wie in Abschnitt 3.3.1 beschrieben, verarbeitet. Damit ist hier eine 1×1-Faltung effektiv eine vollständig verbundene Schicht entlang der spektralen Dimension. Dabei werden für alle Pixel des hyperspektralen Bilds die gleichen Gewichte verwendet. Dadurch bleibt die örtliche Auflösung erhalten. Darüber hinaus ergibt sich der Vorteil, dass das CNN unabhängig von der örtlichen Auflösung der Eingangsdaten ist. Damit können die Testdaten eine andere örtliche Auflösung haben als die Trainingsdaten, was in der Praxis nützlich sein kann. Wird hingegen die Anzahl der Wellenlängenkanäle geändert, muss das Netz neu trainiert werden. Unter Verwendung der gleichen Aufnahmeeinrichtung ist dies jedoch in der Praxis selten notwendig, da Anzahl und mittlere Wellenlänge der Wellenlängenkanäle in der Regel gleich bleiben. Auch im zweiten Teil wird nach einer 1×1-Faltung die *Batch*-Normalisierung durchgeführt, gefolgt von einer logistischen Aktivierungsfunktion. Etwas abweichend dazu werden die letzten Schichten gestaltet.

5.2 Erzwingen der Nebenbedingungen

Die Schichten am Ausgang des CNNs können so gestaltet werden, dass die Nebenbedingungen (2.5) und (2.6) immer erfüllt sind und nicht zusätzlich bei der Minimierung der Verlustfunktion während des Trainings beachtet werden müssen. Dazu wird nach der letzten 1×1-Faltungs-

schicht ebenfalls eine logistische Aktivierungsfunktion verwendet, wodurch die Ausgangsgrößen immer die Nichtnegativitätsbedingung (2.5) erfüllen. Die Normierungsbedingung (2.6) kann mit Hilfe einer zusätzlichen Normierungsschicht

$$a_p = \frac{\tilde{a}_p}{\sum_{\tilde{p}=1}^{P} \tilde{a}_{\tilde{p}}} \quad \text{für} \quad p = 1, 2, \ldots, P \tag{5.1}$$

am Ausgang des CNNs erzwungen werden. Dabei ist $\tilde{a}_p \in \mathbb{R}$ der Wert vor der Normierungsschicht, der mit dem p-ten Reinstoff korrespondiert. Die Normierung wird für jedes Pixel des hyperspektralen Bilds separat durchgeführt, weshalb in (5.1) auf die Ortsindizes verzichtet werden kann. Alternativ hätten beide Nebenbedingungen auch mit Hilfe einer *Softmax*-Schicht

$$a_p = \frac{\mathrm{e}^{\tilde{a}_p}}{\sum_{\tilde{p}=1}^{P} \mathrm{e}^{\tilde{a}_{\tilde{p}}}} \quad \text{für} \quad p = 1, 2, \ldots, P \tag{5.2}$$

erzwungen werden können. Allerdings hat sich gezeigt, dass sich dies negativ auf das Training auswirkt [V4]. Dies liegt vermutlich daran, dass kleine Eingangsänderungen in (5.2) durch die Exponentialfunktionen zu großen Ausgangsänderungen führen, was zu Oszillationen um das Optimum führen kann.

5.3 Einzelne Spektren als Eingangsdaten

Um das CNN für einzelne Spektren als Eingangsdaten zu verwenden, müssen die Dimensionen der Filterkerne der dreidimensionalen Faltungsschichten im ersten Teil entsprechend gewählt werden. Konkret dürfen sie in Richtung der örtlichen Dimensionen nur eine Größe von 1 aufweisen. In spektraler Richtung müssen sie weiterhin eine Größe haben, die größer als 1 ist, da ansonsten keine Faltung stattfindet. Wird dies beachtet, so werden die Spektren unabhängig voneinander, jedoch mit den gleichen Gewichten, verarbeitet. Aufgrund der Netzstruktur ist es daher unerheblich, ob die Spektren als einzelne Datenpunkte oder weiterhin als Hyperspektralbilder eingegeben werden. Wichtig ist dabei,

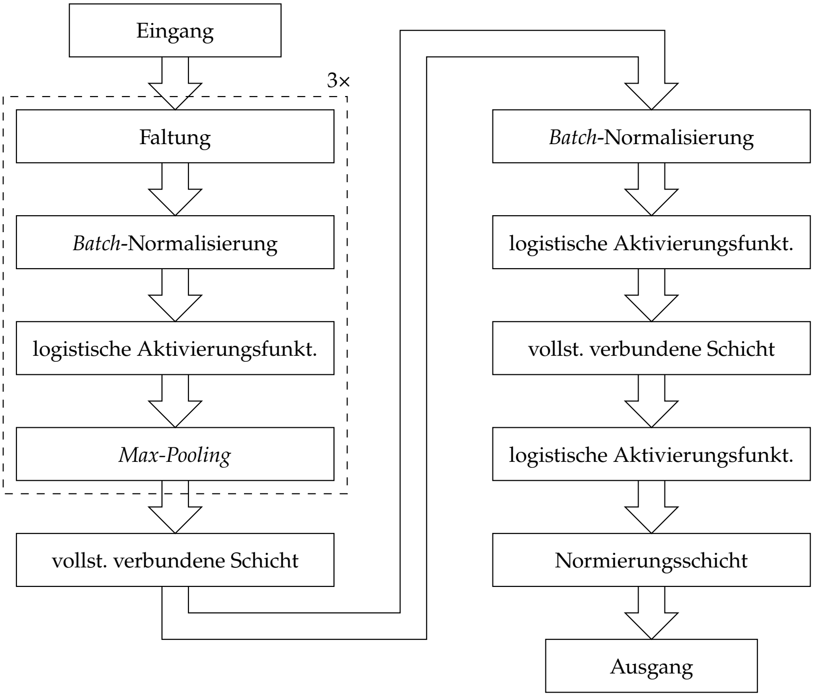


Abbildung 5.3 Netzstruktur des CNNs für die spektrale Entmischung, welche im weiteren Verlauf der Arbeit zur Auswertung der Datenerzeugung und Augmentierung verwendet wird.

dass die Ausgangsdaten das entsprechende Format besitzen. Werden die Spektren als einzelne Datenpunkte behandelt, entsprechen die 1×1-Faltungen im zweiten Teil des CNNs vollständig verbundenen Schichten. Der Aufbau des Netzes mit allen Schichten ist in Abbildung 5.3 zu sehen, wobei hier die Anzahl der sich wiederholenden Blöcke derjenigen entspricht, die im späteren Verlauf dieser Arbeit für die Auswertung verwendet wird. Diese ist in Experimenten als Hyperparameter ermittelt worden (siehe Abschnitt 3.1.3).

Auch Wan et al. [134] nutzen unter Verwendung anderer Trainingsdaten eine ähnliche Netzstruktur, was zeigt, dass diese nicht nur für die in Kapitel 8.1 vorgestellten Datensätze sinnvoll ist. Im Gegensatz zur hier

vorgestellten Netzstruktur werden dort nur zwei Faltungsschichten und eine vollständig verbundene Schichten genutzt.

In den Kapiteln 6 und 7 werden nun Methoden vorgestellt, mit denen die Trainingsdaten für das hier vorgestellte CNN erzeugt bzw. erweitert werden können.

6 Modellbasierte Trainingsdatenerzeugung aus Reinspektren

In diesem Kapitel werden zwei Verfahren zur Erzeugung von Trainingsdaten vorgestellt, die mit wenigen verfügbaren realen Daten auskommen. Je nach verwendetem Mischmodell werden nur die Reinspektren oder zusätzlich einige Mischspektren benötigt. Letztere sind erforderlich, um Parameter bei nichtlinearen Mischmodellen zu ermitteln. Wichtig ist, dass von den Reinspektren jeweils eine Menge verfügbar sein muss, um die Spektrenvariabilität modellieren zu können. Die Menge

$$\mathcal{M}_p = \left\{\mathbf{m}_{p1}, \ldots, \mathbf{m}_{pS}\right\} \tag{6.1}$$

enthält S Spektren des p-ten Reinstoffs. Dieses Szenario ist in einem industriellen Umfeld nicht unüblich, wo die beteiligten Reinstoffe bekannt sind, von denen Spektren als Trainingsdaten aufgenommen werden können. Vorteil ist, dass nur wenige Stoffgemische für die Aufnahme von Mischspektren erzeugt werden müssen.

Die Grundidee ist für beide Verfahren die gleiche: Es werden mit Hilfe von Mischmodellen, die in Abschnitt 2.4 vorgestellt werden, Mischspektren erzeugt, mit denen dann ein KNN für die spektrale Entmischung trainiert werden kann. Die Erzeugung von Trainingsdaten mit Mischmodellen wird bereits von Xu et al. [139] genutzt, jedoch ohne die Modellierung der Spektrenvariabilität. Die hier vorgestellte Vorgehensweise hat zwei Vorteile, verglichen mit der direkten Anwendung der Mischmodelle, die in Abschnitt 2.6 beschrieben wird. Zum einen wird dabei die Spektrenvariabilität modelliert und kann vom KNN berücksichtigt werden. Zum anderen können die Nebenbedingungen (2.5) und (2.6) durch die Netzarchitektur erzwungen werden (siehe Abschnitt 5.2).

Beim ersten Verfahren in Abschnitt 6.1 werden bei der Erzeugung eines Spektrums eines Stoffgemisches die dafür verwendeten Reinspektren zufällig aus den Reinspektrenmengen $\mathcal{M}_p$ ausgewählt. Beim zweiten Verfahren in Abschnitt 6.2 werden die Reinspektren als normalverteilte Zufallsvektoren modelliert, die über ihre Momente beschrieben werden können. Mit Hilfe der Mischmodelle werden die Momente der Mischspektren berechnet, die ebenfalls als normalverteilte Zufallsvektoren beschrieben werden. Trainingsdaten werden daraus durch das Erzeugen von Realisierungen generiert.

6.1 Zufällige Reinspektrenwahl

Das hier vorgestellte Verfahren ist bereits veröffentlicht [V6] worden. Im Unterschied zur Veröffentlichung werden in Kapitel 8 teilweise andere Datensätze zum Training und Test verwendet, damit eine Vergleichbarkeit mit den anderen hier vorgestellten Verfahren gewährleistet ist. Außerdem sind in der Veröffentlichung keine Mischungen in der Albedo-Domäne durchgeführt worden (siehe Abschnitt 2.4.2).

Bei diesem Verfahren wird das CNN für die spektrale Entmischung aus Kapitel 5 mit den künstlich erzeugten Spektren trainiert. Dabei werden die Spektren während des Trainings und nicht im Vorfeld erzeugt. Dadurch wird weniger Speicher (vor allem auf der Grafikkarte) benötigt, in dem andernfalls alle erzeugten Spektren vorgehalten werden müssten. Vorgegeben werden nur die Anteilsvektoren $\mathbf{a}$ und je eine Menge an Reinspektren $\mathcal{M}_p$ für jeden der P enthaltenen Reinstoffe.

Die Anteilsvektoren sind hier also sowohl Eingangs- als auch Ausgangsdaten. Vor der ersten Schicht werden daraus mit Hilfe der Mischmodelle aus Abschnitt 2.4 die Trainingsdaten erzeugt. Die dafür benötigten P Reinspektren $\mathbf{m}_{ps}$ werden für jede Mischung aus den jeweiligen Reinspektrenmengen $\mathcal{M}_p$ zufällig ausgewählt. Dazu wird der Index s für jedes Reinspektrum als Realisierung einer diskreten gleichverteilten Zufallsvariablen $\mathrm{s} \sim \mathcal{U}\{1, S\}$ behandelt.

Um die *Batch*-Größe zu erhöhen, können auch für jeden Anteilsvektor mehrere Spektren in einem Schritt erzeugt werden. Durch das zufällige Ziehen der Reinspektren weisen auch die berechneten Mischspektren eine Spektrenvariabilität auf. Eine Einteilung in Epochen ist hier schwie-

rig, da durch die zufällige Ziehung der Reinspektren das CNN fast immer neue Daten zum Training erhält. Deshalb wird hier eine Epoche als Trainingsabschnitt definiert, bei dem aus allen vorgegebenen Mischverhältnissen einmal Spektren erzeugt werden. Da über viele Epochen trainiert wird, werden dem CNN viele Realisierungen der Spektren für die vorgegebenen Mischverhältnisse präsentiert.

Bei den Trainingsdaten, die mit dem LMM (2.7) erzeugt werden, ergibt die Summe der Vorfaktoren im Mischmodell durch die Normierungsbedingung bereits 1. Dies gilt auch für eine Mischung in der Albedo-Domäne. Dort werden vor der Mischung mit dem LMM alle Reinspektrenmengen mit (2.11) in die Albedo-Domäne überführt. Im Gegensatz dazu ergibt die Summe der Vorfaktoren beim FM, beim GBM und beim LQM nicht 1. Um dies zu erreichen, werden Spektren, die mit FM oder GBM erzeugt werden, mit

$$\left(1 + \sum_{p=1}^{P-1} \sum_{q=p+1}^{P} \gamma_{pq}\, a_p\, a_q\right)^{-1} \tag{6.2}$$

multipliziert und solche, die mit dem LQM erzeugt werden, mit

$$\left(1 + \sum_{p=1}^{P} \sum_{q=1}^{P} \delta_{pq}\right)^{-1} . \tag{6.3}$$

Diese Normierung ist sinnvoll, da das Licht, welches zum linearen Anteil beiträgt, nicht zusätzlich zum bilinearen Anteil beitragen kann [83, 100]. Damit sind für γ_{pq} und δ_{pq} Werte größer als 1 plausibel.

Die Mischmodelle GBM und LQM besitzen Nichtlinearitätskoeffizienten, die vorgegeben werden müssen. In dieser Arbeit wird dafür ein Wert für alle Reinspektrenkombinationen verwendet, was zwar eine Einschränkung darstellt, jedoch in der Praxis gut funktioniert und den Vorteil hat, nicht zu viele Freiheitsgrade einzuführen. So gilt für alle p und q: $\gamma_{pq} = \gamma$ und $\delta_{pq} = \delta$. Für die Bestimmung geeigneter Werte ist ein kleiner Validierungsdatensatz, der aus einigen Spektren von Stoffgemischen besteht, erforderlich.

6.2 Modellierung als normalverteilte Zufallsvektoren

Das in diesem Abschnitt vorgestellte Verfahren ist bereits veröffentlicht worden [V9]. Im Gegensatz zur Veröffentlichung wird in dieser Arbeit die vollständige Herleitung der stochastischen Mischmodelle präsentiert. Darüber hinaus wird für einige Formeln eine abweichende, jedoch äquivalente Darstellung gewählt. Außerdem wird in der Auswertung in Kapitel 8 zusätzlich die Entmischung bzw. Modellierung in der Albedo-Domäne betrachtet.

Wie auch im Verfahren davor werden hier aus Mengen an Reinspektren Mischspektren als Trainingsdaten für ein KNN für die spektrale Entmischung erzeugt. Bei diesem Verfahren geschieht dies jedoch über den Umweg, die Spektrenvariabilität als stochastische Eigenschaften der Reinspektren (und Mischspektren) zu modellieren. Es dient auch dem Zweck, zu prüfen, ob die unten getroffenen Annahmen über die stochastischen Eigenschaften der Spektrenvariabilität zulässig sind oder nicht. Dies kann im Hinblick auf die spektrale Entmischung untersucht werden, indem die Leistung bei der spektralen Entmischung mit anderen Verfahren verglichen wird.

Die Reinspektren $\mathbf{m}_p$ werden als Λ-dimensionale normalverteilte Zufallsvariablen $\mathfrak{m}_p \sim \mathcal{N}\left\{\boldsymbol{\mu}_{\mathfrak{m}_p}, \boldsymbol{\Sigma}_{\mathfrak{m}_p,\mathfrak{m}_p}\right\}$ modelliert. Diese lassen sich mit Hilfe ihres Mittelwertvektors $\boldsymbol{\mu}_{\mathfrak{m}_p} = \boldsymbol{\mu}_p \in \mathbb{R}^{\Lambda}$ und ihrer Autokovarianzmatrix $\boldsymbol{\Sigma}_{\mathfrak{m}_p,\mathfrak{m}_p} = \boldsymbol{\Sigma}_{p,p} \in \mathbb{R}^{\Lambda\times\Lambda}$ beschreiben. Zu Gunsten der Lesbarkeit werden in diesem Kapitel bei Mittelwertvektoren und Kovarianzmatrizen von Reinspektren nur deren Indizes dargestellt, wie bspw. in (6.5) bis (6.8). Die Annahme einer Normalverteilung kann getroffen werden, da die Spektrenvariabilität viele mögliche Ursachen hat und von vielen Freiheitsgraden abhängig ist (siehe Abschnitt 2.5). Damit kann der zentrale Grenzwertsatz in Betracht gezogen werden [32]. Darüber hinaus werden die Reinstoffe als stochastisch unabhängig betrachtet. Diese Annahme ist notwendig, um später die Momente der Mischspektren berechnen zu können. Eine Berechnung der Kreuzkovarianzmatrizen für je zwei Reinstoffe ist nämlich nicht möglich, da die Zuordenbarkeit der einzelnen Spektren nicht gewährleistet ist. Die Annahme der stochastischen Un-

abhängigkeit ist auch plausibel, da die Beschaffenheit der Reinspektren keinen Einfluss auf andere Reinspektren hat.

Die Mischspektren $\boldsymbol{\upsilon}$ werden ebenfalls als Λ-dimensionale normalverteilte Zufallsvariablen $\mathbf{\upsilon} \sim \mathcal{N}\left\{\boldsymbol{\mu}_{\mathbf{\upsilon}}, \boldsymbol{\Sigma}_{\mathbf{\upsilon},\mathbf{\upsilon}}\right\}$ modelliert. Dabei handelt es sich um eine weitere Annahme, da zwar Summen von normalverteilten Zufallsvektoren wieder normalverteilt sind, nicht jedoch Produkte, die für die nichtlinearen Mischmodelle benötigt werden. Die Mittelwertvektoren $\boldsymbol{\mu}_{\mathbf{\upsilon}} \in \mathbb{R}^{\Lambda}$ und Autokovarianzmatrizen $\boldsymbol{\Sigma}_{\mathbf{\upsilon},\mathbf{\upsilon}} \in \mathbb{R}^{\Lambda\times\Lambda}$ der Mischspektren werden mit Hilfe der Rechenregeln des Erwartungswertes berechnet. Die daraus resultierenden Realisierungen stellen die eigentlichen Trainingsdaten dar. Die erforderlichen mathematischen Grundlagen werden in Abschnitt 6.2.1 vorgestellt. In Abschnitt 6.2.2 werden diese auf die Mischmodelle aus Abschnitt 2.4 angewendet.

6.2.1 Mathematische Grundlagen

In diesem Abschnitt wird, auch wenn die Grundlagen allgemeiner Natur sind, die Symbolik der spektralen Entmischung verwendet, wie sie auch in der restlichen Arbeit Verwendung findet.

Die zentrale Rolle spielt der Erwartungswert $\mathrm{E}\{\cdot\}$. Wird dieser, wie hier, auf Vektoren und Matrizen angewendet, so entspricht dies der Anwendung auf die einzelnen Elemente. Der Erwartungswert einer reellen Zufallsvariablen z ist definiert als

$$\mathrm{E}\{\mathrm{z}\} = \int_{-\infty}^{\infty} z f_{\mathrm{z}}(z)\mathrm{d}z\,. \tag{6.4}$$

Dabei ist $f_{\mathrm{z}}(z) \in \mathbb{R} \rightarrow \mathbb{R}$ die Wahrscheinlichkeitsdichtefunktion [59]. Ist diese, wie hier, nicht bekannt, kann der Erwartungswert aus einer Stichprobe geschätzt werden. Der Stichprobenmittelwert eines Zufallsvektors $\mathbf{m}_p$

$$\hat{\boldsymbol{\mu}}_p = \hat{\boldsymbol{\mu}}_{\mathbf{m}_p} = \frac{1}{S}\sum_{s=1}^{S} \mathbf{m}_{ps} \tag{6.5}$$

wird anhand der S Datenpunkte $\mathbf{m}_{ps}$ ermittelt [98]. Darüber hinaus spielt die Kovarianzmatrix

$$\boldsymbol{\Sigma}_{p,q} = \boldsymbol{\Sigma}_{\mathbf{m}_p,\mathbf{m}_q} = \mathrm{Cov}\left\{\mathbf{m}_p, \mathbf{m}_q\right\} = \mathrm{E}\left\{\left(\mathbf{m}_p - \boldsymbol{\mu}_p\right)\left(\mathbf{m}_q - \boldsymbol{\mu}_q\right)^{\mathrm{T}}\right\} \tag{6.6}$$

mit $\boldsymbol{\mu}_p = \boldsymbol{\mu}_{\mathbf{m}_p} = \mathrm{E}\left\{\mathbf{m}_p\right\}$ eine wichtige Rolle [98]. Diese wird für $p = q$ als Autokovarianzmatrix bezeichnet und für $p \neq q$ als Kreuzkovarianzmatrix. Für die Kreuzkovarianzmatrix gilt

$$\boldsymbol{\Sigma}_{p,q} = \boldsymbol{\Sigma}_{q,p}^{\mathrm{T}}\,, \tag{6.7}$$

während die Autokovarianzmatrix symmetrisch ist. Auch die Autokovarianzmatrix kann mit Hilfe von Datenpunkten geschätzt werden [98]:

$$\hat{\boldsymbol{\Sigma}}_{p,p} = \hat{\boldsymbol{\Sigma}}_{\mathbf{m}_p,\mathbf{m}_p} = \frac{1}{S-1}\sum_{s=1}^{S}\left(\mathbf{m}_{ps} - \hat{\boldsymbol{\mu}}_p\right)\left(\mathbf{m}_{ps} - \hat{\boldsymbol{\mu}}_p\right)^{\mathrm{T}}. \tag{6.8}$$

Der Erwartungswert ist ein linearer Operator [98]:

$$\mathrm{E}\left\{\sum_{p=1}^{P} a_p\,\mathbf{m}_p\right\} = \sum_{p=1}^{P} a_p\,\mathrm{E}\left\{\mathbf{m}_p\right\}. \tag{6.9}$$

Aus (6.6) und (6.9) folgt für die Autokovarianzmatrix einer gewichteten Summe von Zufallsvektoren (Herleitung siehe (A.1) im Anhang):

$$\mathrm{Cov}\left\{\sum_{p=1}^{P} a_p\,\mathbf{m}_p, \sum_{p=1}^{P} a_p\,\mathbf{m}_p\right\} = \sum_{p=1}^{P}\sum_{q=1}^{P} a_p\,a_q\,\mathrm{Cov}\left\{\mathbf{m}_p, \mathbf{m}_q\right\}. \tag{6.10}$$

Des Weiteren ergibt sich der Zusammenhang

$$\mathrm{Cov}\left\{\mathbf{m}_p, \mathbf{m}_q\right\} = \mathrm{E}\left\{\mathbf{m}_p\,\mathbf{m}_q^{\mathrm{T}}\right\} - \mathrm{E}\left\{\mathbf{m}_p\right\}\,\mathrm{E}\left\{\mathbf{m}_q\right\}^{\mathrm{T}} \tag{6.11}$$

durch Ausmultiplizieren von (6.6) und aufgrund der Linearität des Erwartungswerts (6.9). Aus (6.11) folgt für das Produkt der skalaren Zufallsvariablen z_1 und z_2

$$\mathrm{E}\{\mathrm{z}_1\,\mathrm{z}_2\} = \mathrm{E}\{\mathrm{z}_1\}\,\mathrm{E}\{\mathrm{z}_2\} + \mathrm{Cov}\{\mathrm{z}_1, \mathrm{z}_2\}\,. \tag{6.12}$$

Dieser Zusammenhang wird nur für skalare Zufallsvariablen benötigt, da in den Mischmodellen die Reinspektren elementweise miteinander multipliziert werden. Sind die Zufallsvariablen unkorreliert, so gilt [98]

$$\mathrm{Cov}\{\mathrm{z}_1, \mathrm{z}_2\} = 0\,. \tag{6.13}$$

Dasselbe gilt auch für unkorrelierte und damit auch für stochastisch unabhängige Zufallsvektoren. Dort verschwindet die Kreuzkovarianzmatrix.

Wichtig für die Herleitung der Modelle im folgenden Abschnitt ist der Zusammenhang

$$(\mathbf{m}_1 \odot \mathbf{m}_2)\,(\mathbf{m}_3 \odot \mathbf{m}_4)^{\mathrm{T}} = \left(\mathbf{m}_1\,\mathbf{m}_3^{\mathrm{T}}\right) \odot \left(\mathbf{m}_2\,\mathbf{m}_4^{\mathrm{T}}\right), \tag{6.14}$$

wobei $\mathbf{m}_1$ bis $\mathbf{m}_4$ beliebige reelle Vektoren der gleichen Länge sind. Der Zusammenhang wird klar, wenn das Assoziativgesetz auf die Elemente der resultierenden Matrix angewendet wird.

Verallgemeinerter Satz von Isserlis

Ebenfalls wichtig für die Herleitung ist der verallgemeinerte Satz von Isserlis [56, 137]. Sei $\mathbf{m} \sim \mathcal{N}\left\{\mathbf{0}, \mathbf{\Sigma}_{\mathbf{m},\mathbf{m}}\right\} \in \mathbb{R}^{\Lambda}$ ein zentrierter Normalverteilter Zufallsvektor und $\mathcal{L} = \{\lambda_1, \ldots, \lambda_B\}$ mit $1 \leq \lambda_b \leq \Lambda$ eine Menge an B Indizes, wobei Indizes auch mehrfach vorkommen dürfen, so gilt für ungerade B:

$$\mathrm{E}\left\{\prod_{\lambda_b \in \mathcal{L}} \mathrm{m}_{\lambda_b}\right\} = 0\,. \tag{6.15}$$

Ist B gerade, kann die Menge $\mathcal{L}$ in eine Paar-Partition überführt werden. Dabei wird die Menge $\mathcal{L}$ in Untermengen aufgeteilt, die jeweils aus genau 2 Elementen bestehen. Es sei $\mathfrak{B}(\mathcal{L})$ der Raum aller möglichen Paar-Partitionen von $\mathcal{L}$. Seien $\mathcal{L}^{\mathrm{PP}} \in \mathfrak{B}(\mathcal{L})$ eine mögliche Paar-Partition und $\mathcal{L}^{\mathrm{P}} \in \mathcal{L}^{\mathrm{PP}}$ ein Paar an Indizes, wobei das erste Element eines Paares stets mit λ_{A} und das zweite mit λ_{B} bezeichnet wird, dann gilt für gerade B:

$$\mathrm{E}\left\{\prod_{\lambda_b \in \mathcal{L}} \mathrm{m}_{\lambda_b}\right\} = \sum_{\mathcal{L}^{\mathrm{PP}} \in \mathfrak{B}(\mathcal{L})} \prod_{\substack{\mathcal{L}^{\mathrm{P}} \in \mathcal{L}^{\mathrm{PP}} \\ \lambda_{\mathrm{A}}, \lambda_{\mathrm{B}} \in \mathcal{L}^{\mathrm{P}}}} \mathrm{Cov}\left\{\mathrm{m}_{\lambda_{\mathrm{A}}}, \mathrm{m}_{\lambda_{\mathrm{B}}}\right\}. \tag{6.16}$$

In den Herleitungen zu der hier vorgestellten Methode wird der verallgemeinerte Satz von Isserlis für Produkte von Zufallsvektoren mit sich selbst benötigt. Da die Produkte entweder elementweise sind oder ein Vektor mit einem Zeilenvektor multipliziert wird (oder einer Kombination aus beidem), sind die Elemente des resultierenden Vektors bzw. der resultierenden Matrix Produkte der Elemente des Zufallsvektors. So

kann der Satz auf die jeweiligen Elemente angewandt werden. Auf diese Weise ist er auch hilfreich für den ganzen Vektor bzw. die ganze Matrix.

6.2.2 Stochastische Mischmodelle

In diesem Abschnitt werden die stochastischen Versionen der Mischmodelle basierend auf Reinstoffvektoren, die als normalverteilte Zufallsvektoren modelliert werden, vorgestellt. In der Praxis werden dazu für jeden Reinstoff der Mittelwertvektor und die Autokovarianzmatrix mit Hilfe von (6.5) und (6.8) geschätzt. Zu Gunsten der Lesbarkeit wird hier jedoch in den Formeln auf die Symbolik der Schätzung verzichtet, was an den Zusammenhängen nichts ändert. Ziel ist es, für jedes Mischmodell Mittelwertvektor und Autokovarianzmatrix in Abhängigkeit der Mittelwertvektoren sowie der Autokovarianzmatrizen der Reinstoffe darzustellen. Analog zu Abschnitt 6.1 gilt für die Nichtlinearitätskoeffizienten von GBM und LQM für alle p und q: $\gamma_{pq} = \gamma$ und $\delta_{pq} = \delta$.

Bei der stochastischen Version des LMMs

$$\mathbf{\upsilon} = \sum_{p=1}^{P} a_p \, \mathbf{m}_p \tag{6.17}$$

folgt die Formel für den Mittelwertvektor direkt aus (6.9)

$$\boldsymbol{\mu}_{\mathbf{\upsilon}} = \sum_{p=1}^{P} a_p \, \boldsymbol{\mu}_p \tag{6.18}$$

und die Formel für die Autokovarianzmatrix aus (6.10)

$$\boldsymbol{\Sigma}_{\mathbf{\upsilon},\mathbf{\upsilon}} = \sum_{p=1}^{P} a_p^2 \, \boldsymbol{\Sigma}_{p,p} \,, \tag{6.19}$$

wobei aufgrund der Unkorreliertheit der Reinspektren sämtliche Kreuzkovarianzmatrizen in (6.10) verschwinden. Bei der Mischung der Spektren in der Albedo-Domäne wird das LMM für die mit (2.11) transformierten Spektren genutzt, weshalb hier die gleichen Formeln verwendet werden können.

Beim FM (entspricht GBM mit $\gamma = 1$) und GBM werden Reinspektren auch miteinander multipliziert, jedoch nie mit sich selbst:

$$\mathbf{\upsilon} = \sum_{p=1}^{P} a_p \, \mathbf{m}_p + \sum_{p=1}^{P-1} \sum_{q=p+1}^{P} \gamma \, a_p \, a_q \, \mathbf{m}_p \odot \mathbf{m}_q \, . \tag{6.20}$$

Die Formel für den Mittelwertvektor folgt hier direkt aus (6.9), (6.12) und (6.13):

$$\boldsymbol{\mu}_{\mathbf{\upsilon}} = \sum_{p=1}^{P} a_p \, \boldsymbol{\mu}_p + \sum_{p=1}^{P-1} \sum_{q=p+1}^{P} \gamma \, a_p \, a_q \, \boldsymbol{\mu}_p \odot \boldsymbol{\mu}_q \, . \tag{6.21}$$

Die Formel für die Autokovarianzmatrix folgt beim GBM aus (6.10) und (6.13):

$$\begin{aligned}
\boldsymbol{\Sigma}_{\mathbf{\upsilon},\mathbf{\upsilon}} = & \sum_{p=1}^{P} a_p^2 \, \boldsymbol{\Sigma}_{p,p} + \sum_{p=1}^{P-1} \sum_{q=p+1}^{P} \gamma^2 \, a_p^2 \, a_q^2 \, \boldsymbol{\Sigma}_{p\odot q, p\odot q} && \text{(6.22a)} \\
& + \sum_{p=1}^{P} \sum_{\substack{q=1 \\ q \neq p}}^{P} \gamma \, a_p^2 \, a_q \left(\boldsymbol{\Sigma}_{p, p\odot q} + \boldsymbol{\Sigma}_{p\odot q, p} \right) && \text{(6.22b)} \\
& + \sum_{p=1}^{P} \sum_{\substack{q=1 \\ q \neq p}}^{P-1} \sum_{\substack{r=q+1 \\ r \neq p}}^{P} \gamma^2 \, a_p^2 \, a_q \, a_r \left(\boldsymbol{\Sigma}_{p\odot q, p\odot r} + \boldsymbol{\Sigma}_{p\odot r, p\odot q} \right) . && \text{(6.22c)}
\end{aligned}$$

Dabei finden sich in Zeile (6.22a) die Autokovarianzmatrizen der linear und der bilinear eingehenden Reinspektren. Darunter in (6.22b) befinden sich die Kreuzkovarianzmatrizen der linear eingehenden Spektren und der bilinear eingehenden Spektren, die den entsprechenden Reinstoff ebenfalls beinhalten. In (6.22c) befinden sich schließlich die Kreuzkovarianzmatrizen der bilinear eingehenden Spektren, bei denen jeweils ein Reinstoff der gleiche ist. Alle weiteren Kovarianzmatrizen verschwinden aufgrund der stochastisch unabhängigen Reinstoffe.

Die Zusammenhänge zwischen den Kovarianzmatrizen in (6.22a) bis (6.22c) sind in (6.23) bis (6.25) dargestellt, wobei **1** ein Vektor der passen-

den Länge bestehend aus Einsen ist. Die dazugehörigen Herleitungen (A.2), (A.3) und (A.4) sind im Anhang zu finden.

$$\boldsymbol{\Sigma}_{p\odot q,p\odot q} = \boldsymbol{\Sigma}_{p,p} \odot \boldsymbol{\Sigma}_{q,q} + \boldsymbol{\Sigma}_{p,p} \odot \left(\boldsymbol{\mu}_q \boldsymbol{\mu}_q^{\mathrm{T}}\right) + \boldsymbol{\Sigma}_{q,q} \odot \left(\boldsymbol{\mu}_p \boldsymbol{\mu}_p^{\mathrm{T}}\right) \tag{6.23}$$

$$\boldsymbol{\Sigma}_{p,p\odot q} + \boldsymbol{\Sigma}_{p\odot q,p} = \boldsymbol{\Sigma}_{p,p} \odot \left(\mathbf{1}\, \boldsymbol{\mu}_q^{\mathrm{T}}\right) + \boldsymbol{\Sigma}_{p,p} \odot \left(\boldsymbol{\mu}_q\, \mathbf{1}^{\mathrm{T}}\right) \tag{6.24}$$

$$\boldsymbol{\Sigma}_{p\odot q,p\odot r} + \boldsymbol{\Sigma}_{p\odot r,p\odot q} = \boldsymbol{\Sigma}_{p,p} \odot \left(\boldsymbol{\mu}_q \boldsymbol{\mu}_r^{\mathrm{T}}\right) + \boldsymbol{\Sigma}_{p,p} \odot \left(\boldsymbol{\mu}_r \boldsymbol{\mu}_q^{\mathrm{T}}\right) \tag{6.25}$$

Die stochastische Version des LQMs lässt sich so umschreiben, dass sie weitestgehend dem GBM mit einem zusätzlichen quadratischen Summanden, bei dem Reinspektren mit sich selbst multipliziert werden, entspricht:

$$\boldsymbol{\upsilon} = \sum_{p=1}^{P} a_p\, \mathbf{m}_p + \sum_{p=1}^{P} \sum_{q=1}^{P} \delta\, \mathbf{m}_p \odot \mathbf{m}_q \tag{6.26a}$$

$$= \sum_{p=1}^{P} a_p\, \mathbf{m}_p + \delta\, \mathbf{m}_p \odot \mathbf{m}_p + \sum_{p=1}^{P-1} \sum_{q=p+1}^{P} 2\, \delta\, \mathbf{m}_p \odot \mathbf{m}_q\,. \tag{6.26b}$$

Der Mittelwertvektor wird beim LQM mit

$$\boldsymbol{\mu}_{\boldsymbol{\upsilon}} = \sum_{p=1}^{P} a_p\, \boldsymbol{\mu}_p + \delta\, \boldsymbol{\mu}_{p\odot p} + \sum_{p=1}^{P-1} \sum_{q=p+1}^{P} 2\, \delta\, \boldsymbol{\mu}_p \odot \boldsymbol{\mu}_q \tag{6.27}$$

berechnet. Dabei wird der Mittelwert des Produkts eines Reinspektrums mit sich selbst (2. Summand) aufgrund von (6.12) folgendermaßen berechnet:

$$\boldsymbol{\mu}_{p\odot p} = \boldsymbol{\mu}_p \odot \boldsymbol{\mu}_p + d\left(\boldsymbol{\Sigma}_{p,p}\right)\,. \tag{6.28}$$

Für die Autokovarianzmatrix des Mischspektrums müssen wegen der zusätzlichen Summanden im Vergleich zum GBM mehr Kreuzkovarianzmatrizen berechnet werden:

$$\boldsymbol{\Sigma}_{\mathbf{\upsilon},\mathbf{\upsilon}} = \sum_{p=1}^{P} a_p^2 \, \boldsymbol{\Sigma}_{p,p} + \delta^2 \, \boldsymbol{\Sigma}_{p \odot p, p \odot p} + \sum_{p=1}^{P-1} \sum_{q=p+1}^{P} 4 \, \delta^2 \, \boldsymbol{\Sigma}_{p \odot q, p \odot q} \tag{6.29a}$$

$$+ \sum_{p=1}^{P} a_p \, \delta \left(\boldsymbol{\Sigma}_{p \odot p, p} + \boldsymbol{\Sigma}_{p, p \odot p} \right) \tag{6.29b}$$

$$+ \sum_{p=1}^{P} \sum_{\substack{q=1 \\ q \neq p}}^{P} 2 \, a_p \, \delta \left(\boldsymbol{\Sigma}_{p, p \odot q} + \boldsymbol{\Sigma}_{p \odot p, p} \right) \tag{6.29c}$$

$$+ \sum_{p=1}^{P} \sum_{\substack{q=1 \\ q \neq p}}^{P} 2 \, \delta^2 \left(\boldsymbol{\Sigma}_{p \odot p, p \odot q} + \boldsymbol{\Sigma}_{p \odot q, p \odot p} \right) \tag{6.29d}$$

$$+ \sum_{p=1}^{P} \sum_{\substack{q=1 \\ q \neq p}}^{P-1} \sum_{\substack{r=q+1 \\ r \neq p}}^{P} 4 \, \delta^2 \left(\boldsymbol{\Sigma}_{p \odot q, p \odot r} + \boldsymbol{\Sigma}_{p \odot r, p \odot q} \right). \tag{6.29e}$$

In (6.29a) sind die Autokovarianzmatrizen aller Summanden zu finden. Die Zeilen (6.29c) und (6.29e) entsprechen denen, die auch beim GBM vorkommen. Die Kreuzkovarianzmatrizen in (6.29b) und (6.29d) kommen durch die Produkte der Spektren mit sich selbst hinzu. Während erstere in Kombination mit dem gleichen linearen Spektrum zustande kommen, kommen letztere durch die Kombination mit den bilinearen Summanden hinzu, bei denen ein Spektrum dem eines quadratisch eingehenden entspricht.

Die in (6.29a) bis (6.29e) vorkommenden Kovarianzmatrizen, die nicht in (6.23) bis (6.25) berechnet werden, werden in (6.30) bis (6.32) berechnet, wobei **1** ein Vektor der passenden Länge bestehend aus Einsen ist. Die

dazugehörigen Herleitungen (A.5), (A.6), (A.7) und (A.8) sind im Anhang zu finden.

$$\mathbf{\Sigma}_{p\odot p,p} + \mathbf{\Sigma}_{p,p\odot p} = 2\,\mathbf{\Sigma}_{p,p} \odot \left(\boldsymbol{\mu}_p\,\mathbf{1}^{\mathrm{T}}\right) + 2\,\mathbf{\Sigma}_{p,p} \odot \left(\mathbf{1}\,\boldsymbol{\mu}_p^{\mathrm{T}}\right) \tag{6.30}$$

$$\mathbf{\Sigma}_{p\odot p,p\odot q} + \mathbf{\Sigma}_{p\odot q,p\odot p} = 2\,\mathbf{\Sigma}_{p,p} \odot \left(\boldsymbol{\mu}_p\boldsymbol{\mu}_q^{\mathrm{T}}\right) + 2\,\mathbf{\Sigma}_{p,p} \odot \left(\boldsymbol{\mu}_q\boldsymbol{\mu}_p^{\mathrm{T}}\right) \tag{6.31}$$

$$\mathbf{\Sigma}_{p\odot p,p\odot p} = 4\,\mathbf{\Sigma}_{p,p} \odot \left(\boldsymbol{\mu}_p\boldsymbol{\mu}_p^{\mathrm{T}}\right) + 2\,\mathbf{\Sigma}_{p,p} \odot \mathbf{\Sigma}_{p,p} \tag{6.32}$$

Nun können für alle in Abschnitt 2.4 vorgestellten Mischmodelle Mittelwertvektoren und Autokovarianzmatrizen für Mischspektren erzeugt werden. Dazu können verschiedene Anteilsvektoren vorgegeben werden. Für jeden Anteilsvektor werden Mittelwertvektor und Autokovarianzmatrix vorab berechnet. Während des Trainings des CNNs aus Kapitel 5 werden daraus in jeder Epoche für alle Mittelwertvektoren und Autokovarianzmatrizen mehrere unterschiedliche Spektren zufällig erzeugt.

Beim LQM und GBM werden die Nichtlinearitätskoeffizienten γ und δ auch hier mit Hilfe der Performanz des CNNs bezüglich des Validierungsdatensatzes ermittelt.

Analog zum Verfahren in Abschnitt 6.1 werden auch hier die resultierenden Spektren so normiert, dass die Summe aller Vorfaktoren 1 ergibt. Dazu können ebenfalls (6.2) und (6.3) verwendet werden.

7 Augmentierung spektraler Datensätze

Sind neben den Mengen an Reinspektren (6.1) auch Mengen an Spektren von Stoffgemischen vorhanden, können diese als Trainingsdaten für ein CNN für die spektrale Entmischung (siehe Kapitel 5) verwendet werden. Die Menge

$$Y_{\mathbf{a}} = \left\{ \boldsymbol{v}_{\mathbf{a}1}, \ldots, \boldsymbol{v}_{\mathbf{a}S} \right\} \tag{7.1}$$

enthält S Spektren eines Stoffgemisches, in dem die Anteile der Reinstoffe den Elementen des Anteilsvektors $\mathbf{a}$ entsprechen. Da diese, vor allem bei speziellen industriellen Anwendungen, häufig nur in geringer Menge zur Verfügung stehen bzw. aufwendig zu erzeugen sind, stellt das Training eines neuronalen Netzes eine Herausforderung dar (siehe Abschnitt 3.1). Vor allem bei einem Regressionsproblem mit kontinuierlichen Ein- und Ausgangsgrößen ist es wünschenswert, wenn der Eingangswertebereich und der Ausgangswertebereich möglichst gut durch die Daten abgedeckt werden. Übertragen auf die spektrale Entmischung bedeutet das, dass zu möglichst vielen Anteilsvektoren Spektren verfügbar sind. Diese sollten eine möglichst realistische Spektrenvariabilität aufweisen, damit diese durch das KNN berücksichtigt wird.

Abhilfe kann hier die Datenaugmentierung (siehe 3.1.3) schaffen. Jedoch kommen die gängigen Augmentierungsverfahren vor allem aus dem Bereich der Bildverarbeitung und können nicht direkt auf Spektren angewendet werden. Bei der spektralen Entmischung ist die Erzeugung von Trainingsdaten für Anteilsvektoren sinnvoll, die nicht bereits im vorhandenen Trainingsdatensatz enthalten sind. Um die Spektrenvariabilität zu berücksichtigen, können viele unterschiedliche Spektren je Anteilsvektor erzeugt werden. In diesem Kapitel werden Verfahren vorgestellt, die aus relativ wenigen verfügbaren Datenpunkten den zugrundeliegenden Zusammenhang zwischen Anteilsvektoren und Spektren modellieren.

Die Spektrenvariabilität wird dabei über stochastische Bestandteile modelliert. Bisher existieren nur Verfahren, die zusätzliche Reinspektren erzeugen (siehe Abschnitt 4.3 und [8, 9]).

Allen vorgestellten Verfahren ist gemein, dass sie nach dem Training in der Lage sind, unter Vorgabe von Anteilsvektoren Spektren zu erzeugen. Im Detail unterscheiden sich die Verfahren jedoch. In Abschnitt 7.1 wird ein generatives CNN vorgestellt, bei dem neben dem Anteilsvektor zufällige Eingangswerte vorgegeben werden können, um die Spektrenvariabilität zu berücksichtigen. Dieses CNN wird in Abschnitt 7.2 in ein GAN integriert. Ein davon unabhängiger Ansatz wird in Abschnitt 7.3 vorgestellt. Hier werden die Spektren als Gauß-Prozesse modelliert. Die Zusammenhänge zwischen den Momenten der Reinspektren und der Spektren von Mischungen werden mit Hilfe von KNN hergestellt, die kleiner sind als die der vorherigen Ansätze.

7.1 Generatives Faltungsnetz

Der in diesem Abschnitt vorgestellte Ansatz ist bereits veröffentlicht worden [V7]. Dabei fällt in dieser Arbeit die Auswertung in Kapitel 8 ausführlicher aus. Vor allem wird zusätzlich die Entmischung bzw. Augmentierung in der Albedo-Domäne betrachtet.

Die Grundidee dieses Ansatzes besteht darin, ein CNN zu trainieren, bei dem die Ein- und Ausgänge im Vergleich zum CNN aus Kapitel 5 vertauscht sind. Damit sind die Anteilsvektoren die Eingangsgrößen und die Spektren die Ausgangsgrößen. So lassen sich mit dem fertig trainierten generativen CNN unter Vorgabe neuer Anteilsvektoren neue Spektren erzeugen. Ein trainiertes KNN würde jedoch für jeden vorgegebenen Anteilsvektor $\mathbf{a}$ genau ein Spektrum erzeugen, da es einer Funktion entspricht. Hier kommen die zufälligen Eingangsgrößen $\mathbf{z} \in \mathbb{R}^Z$ ins Spiel. Werden zusammen mit den Anteilsvektoren unterschiedliche Realisierungen der zufälligen Eingangsgrößen vorgegeben, kann das generative CNN unterschiedliche Spektren für jeden Anteilsvektor erzeugen.

Da für das Training des generativen CNNs die gleichen Trainingsdaten zur Verfügung stehen wie für das CNN für die spektrale Entmischung, stellt sich die Frage, wie durch den Umweg über das generative CNN und die Erzeugung zusätzlicher Trainingsdaten eine Verbesserung er-

zielt werden kann. Zentral ist hier die Spektrenvariabilität. Durch sie haben jeweils mehrere Spektren den gleichen Anteilsvektor. Dadurch gibt es beim CNN für die spektrale Entmischung wesentlich mehr unterschiedliche Eingangsdaten als Ausgangsdaten. Beim generativen CNN verhält es sich jedoch durch das Vertauschen von Ein- und Ausgangsdaten genau umgekehrt. Durch die zufälligen Eingangsgrößen wird die Anzahl an Eingangsdaten künstlich erhöht. Um den Raum, der durch die zufälligen Eingangsgrößen aufgespannt wird, möglichst gut abdecken zu können, werden diese in jeder Epoche beim Training neu erzeugt. Vor allem im industriellen Umfeld ist ein Datensatz plausibel, bei dem viele unterschiedliche Spektren je Anteilsvektor enthalten sind (siehe Abschnitt 1.2).

Wie in Abschnitt 2.5 dargestellt, hat die Spektrenvariabilität viele Ursachen. Um sie korrekt modellieren zu können, müssen die Bedingungen, unter denen ein Spektrum aufgenommen wird, genau bekannt sein. Dies ist meist nicht der Fall. Eine mögliche Interpretation der zufälligen Eingangsgrößen ist, dass sie die von einem KNN kodierte Version der Umgebungsparameter, die für die spektrale Entmischung verantwortlich sind, darstellen. Eine gängige Netzstruktur, die diese Kodierung durchführen könnte, wäre ein *Autoencoder* (siehe Abschnitt 3.5). Die Daten in der Engstelle entsprächen der Kodierung. Da die nötigen Umgebungsparameter zum Training des *Autoencoders* nicht verfügbar sind, kann dieser nicht realisiert werden, jedoch kann diese Interpretation hilfreich sein, zu verstehen, weshalb die zufälligen Eingangsgrößen zum gewünschten Ergebnis führen. Es wäre ein *Autoencoder* denkbar, der durch Training genau diese Größen als Kodierung der Umgebungsparameter erhält.

7.1.1 Netzaufbau

Der Aufbau des generativen CNNs wird in Abbildung 7.1 dargestellt. Es besteht im Wesentlichen aus Faltungsschichten, die eine sogenannte transponierte Faltung durchführen. Bei mehreren Ein- und Ausgangsmerkmalskarten wird analog zu den Abbildungen 3.4 und 3.5 verfahren. Die Parameter der transponierten Faltung sind hier so gewählt, dass ihr Resultat dem Einfügen von Nullen zwischen den Elementen des Eingangsvektors mit anschließender Faltung entspricht.

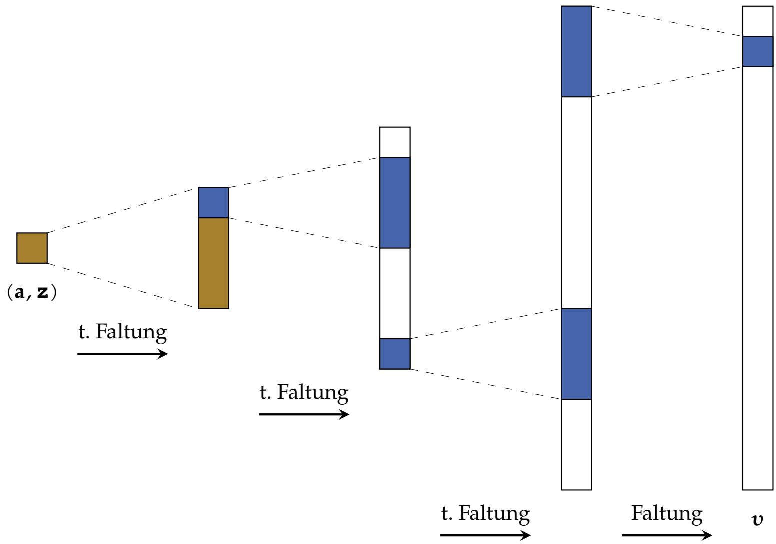


Abbildung 7.1 Das generative CNN, wie es für die Ergebnisse in dieser Arbeit verwendet wird [V7]. Es werden drei Schichten mit transponierten Faltungen (t. Faltung) benutzt, um aus den Anteilsvektoren und zufälligen Eingängen Spektren zu erzeugen. Die letzte Schicht ist eine Faltungsschicht, die alle Merkmalskarten der Vorgängerschicht zusammenführt. Zu Gunsten der Lesbarkeit wird in der Abbildung immer nur eine Merkmalskarte je Schicht dargestellt.

In jeder Schicht wird die logistische Funktion (3.7) als Aktivierungsfunktion verwendet. Diese hat in den Versuchen zu den besten Ergebnissen geführt. So gibt es durch die Begrenzung der Ausgangswerte im Vergleich zur ReLU weniger Ausreißer bei den generierten Spektren. Dies gilt besonders für das Verfahren in Abschnitt 7.1.3.1.

Als Verlustfunktion wird die logistische Verlustfunktion (3.4) verwendet, wobei die Spektren die Ausgangsdaten sind. Diese Verlustfunktion wird hauptsächlich für KNN verwendet, die eine binäre Klassifikation durchführen. Sie kann aber auch für kontinuierliche Ausgangsgrößen eingesetzt werden, so lange sie immer zwischen 0 und 1 liegen, was hier der Fall ist, da die Spektren einem Weißabgleich unterzogen werden (siehe Abschnitt 8.1).

7.1.2 Interpretation als Conditional Variational Autoencoder

Das generative CNN kann alternativ auch als *Decoder* eines *Conditional Variational Autoencoders* (siehe Abschnitt 3.5) interpretiert werden. Im Falle des hier vorgestellten generativen CNNs sind die Bedingungen die vorgegebenen Materialanteile. Während der Versuche hat sich jedoch herausgestellt, dass die Ergebnisse identisch sind, unabhängig davon, ob der *Encoder* verwendet wird oder, wie oben beschrieben, direkt zufällige Werte basierend auf einer festen Wahrscheinlichkeitsverteilung vorgegeben werden. Das liegt daran, dass beim *Variational Autoencoder* die Kostenfunktion nicht nur die Rekonstruktion der Eingangsdaten beinhaltet, sondern auch einen Teil, der die Ausgangsgrößen des *Encoders* mit denen einer vorgegebenen Wahrscheinlichkeitsverteilung vergleicht (siehe Abschnitt 3.5). Bei sich unterscheidenden Daten im Trainingsdatensatz ist es nicht möglich, dass die vorgegebene Wahrscheinlichkeitsverteilung beim Training erreicht wird und gleichzeitig die Rekonstruktion einen möglichst geringen Fehler erreicht. Hier werden jedoch die Anteilsvektoren vorgegeben, weshalb ein großer Teil der Information direkt in den *Decoder* gegeben wird. Für die Spektrenvariabilität scheint es ausreichend zu sein, direkt die gewünschte Wahrscheinlichkeitsverteilung vorzugeben. Diese wird während des Trainings vom *Encoder* immer erreicht, sodass dessen Training auch ganz entfallen kann.

7.1.3 Verbesserung der modellierten Spektrenvariabilität

Während der Umsetzung hat sich herausgestellt, dass die Spektrenvariabilität der generierten Spektren weniger ausgeprägt ist als die der Daten im entsprechenden Trainingsdatensatz (siehe Abschnitt 8.2). Das liegt daran, dass das generative CNN lange trainiert wird und sich die meisten Spektren in der Nähe des (jeweils für einen bestimmten Anteilsvektor) mittleren Spektrums befinden. Da gleichzeitig die zufälligen Eingänge in jeder Epoche und für jeden Datenpunkt neue Realisierungen liefern, kommen Spektren mit größerem Abstand zum mittleren Spektrum bei der Datenerzeugung kaum vor. Um die Spektrenvariabilität besser modellieren zu können, werden zwei Ergänzungen vorgestellt.

7.1.3.1 Varianzvariation der zufälligen Eingänge

Bei der ersten Erweiterung wird die Varianz der zufälligen Eingangsgrößen bei der Erzeugung zusätzlicher Daten größer gewählt als beim Training des generativen CNNs. Dies führt zu einer höheren Varianz der erzeugten Spektren. Durch die Verwendung logistischer Aktivierungsfunktionen ist der Wertebereich der Ausgangsgrößen jeder Schicht beschränkt, sodass auch bei unbekannten zufälligen Eingangswerten keine extremen Ausreißer zu erwarten sind. Jedoch besteht die Gefahr, dass die erzeugten Spektren, bei zu stark von den aus dem Training bekannten Eingangswerten abweichenden Eingangsgrößen, nicht mehr authentisch sind. Daher darf die Varianz der zufälligen Eingänge nicht zu stark erhöht werden.

7.1.3.2 Regularisierung mit Kovarianzmatrizen

Bei der zweiten Erweiterung wird eine Regularisierung basierend auf Autokovarianzmatrizen verwendet. Die Voraussetzung dafür ist, dass genügend Spektren mit (näherungsweise) identischem Anteilsvektor zur Verfügung stehen.

Für die Regularisierung wird für jede im Trainingsdatensatz enthaltene Menge an Spektren $Y_{\mathbf{a}}$ ein Regularisierungsterm

$$\mathrm{k}_{\boldsymbol{\Sigma}} \left\| \hat{\boldsymbol{\Sigma}}_{Y_{\mathbf{a}},Y_{\mathbf{a}}} - \hat{\boldsymbol{\Sigma}}_{Y^*_{\mathbf{a}},Y^*_{\mathbf{a}}} \right\|_{\mathrm{F}}^2 \tag{7.2}$$

zur logistischen Verlustfunktion (3.4) addiert. Der Gewichtungsfaktor $\mathrm{k}_{\boldsymbol{\Sigma}} \in \mathbb{R}^+$ ist dabei für alle Summanden gleich. Die Schätzung der Autokovarianzmatrizen mit der Menge an erzeugten Spektren $\hat{\boldsymbol{\Sigma}}_{Y_{\mathbf{a}},Y_{\mathbf{a}}}$ und mit der Menge an wahren Spektren $\hat{\boldsymbol{\Sigma}}_{Y^*_{\mathbf{a}},Y^*_{\mathbf{a}}}$ erfolgt mit Hilfe von

$$\hat{\boldsymbol{\mu}}_{Y_{\mathbf{a}}} = \frac{1}{S} \sum_{s=1}^{S} \boldsymbol{v}_{\mathbf{a}s} \tag{7.3}$$

und

$$\hat{\boldsymbol{\Sigma}}_{Y_{\mathbf{a}},Y_{\mathbf{a}}} = \frac{1}{S-1} \sum_{s=1}^{S} \left(\boldsymbol{v}_{\mathbf{a}s} - \hat{\boldsymbol{\mu}}_{Y_{\mathbf{a}}} \right) \left(\boldsymbol{v}_{\mathbf{a}s} - \hat{\boldsymbol{\mu}}_{Y_{\mathbf{a}}} \right)^{\mathrm{T}}. \tag{7.4}$$

Dadurch, dass hier auch die Autokovarianzmatrizen, die nur aufgrund der Spektrenvariabilität existieren, mit in die Verlustfunktion fließen, wird eine falsche Modellierung der Spektrenvariabilität bestraft.

7.2 Generative Adversarial Network

Das in diesem Abschnitt vorgestellte Verfahren ist bereits veröffentlicht worden [V10], wobei in dieser Arbeit die Auswertung in Kapitel 8 ausführlicher ausfällt. Vor allem die Augmentierung in der Albedo-Domäne und die spektrale Entmischung werden in der Auswertung in Kapitel 8 zusätzlich betrachtet.

Wie bereits in den Grundlagen in Abschnitt 3.4 beschrieben, eignen sich GAN dazu, Daten zu erzeugen, die den verwendeten Trainingsdaten ähnlich sind. Um sie, wie das generative CNN aus Abschnitt 7.1, für die Erzeugung weiterer Mischspektren mit neuen Anteilsvektoren einsetzen zu können, müsste es eine Möglichkeit geben, die Anteilsvektoren vorzugeben. Dies ist beim GAN so zunächst nicht möglich (siehe Abbildung 3.7), wobei die realen Daten in Abbildung 3.7 die Spektren des Trainingsdatensatzes wären.

Eine Abwandlung des GANs, die zusätzliche Eingangsgrößen berücksichtigt, ist das *conditional Generative Adversarial Network* (CGAN) [80], was in Abbildung 7.2 schematisch für die hier verwendeten Daten dargestellt wird. Das CGAN ist entwickelt worden, um bei der Datenerzeugung (sowie beim Training) Klassen vorgeben zu können. Damit wird die Erzeugung von Daten ermöglicht, die der entsprechenden Klasse zugehören. Im Gegensatz zu den vorgegebenen Anteilsvektoren ist die Anzahl der Klassen endlich. Außerdem kommen auch alle Klassen im Trainingsdatensatz vor. Das sind neben der zu geringen Anzahl an Trainingsdaten die Gründe dafür, warum es in den Versuchen zu diesem Ansatz nicht gelungen ist, ein CGAN für die Erzeugung zusätzlicher Spektren zu trainieren. Daher konnten auch keine Ergebnisse für einen Vergleich erzeugt werden. Auch die Verwendung eines *continuous conditional Generative Adversarial Networks* [21] führte nicht zu Erfolg, was ebenfalls an der geringen Anzahl an Trainingsdaten und an dem großen Abstand zwischen den im Trainingsdatensatz vorhandenen Anteilsvek-

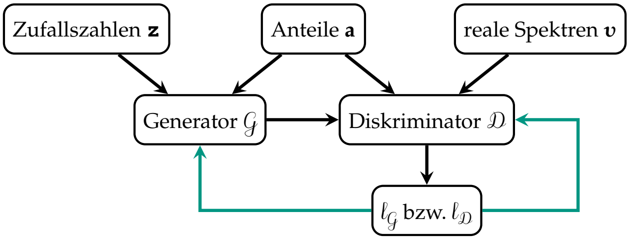


Abbildung 7.2 Schematische Darstellung eines CGANs für die Erzeugung zusätzlicher Mischspektren. Schwarze Pfeile stehen für den Datenfluss in Vorwärtsrichtung, grüne für die Fehlerrückführung. Im Gegensatz zum GAN können hier noch Bedingungen (die Anteile) vorgegeben werden, die sowohl in den Generator als auch in den Diskriminator eingehen.

toren liegt. Außerdem ist es ursprünglich für eine skalare Bedingung und nicht für eine vektorwertige entworfen worden.

Aufgrund dieser Erkenntnisse wird hier das generative CNN aus Abschnitt 7.1 in ein GAN integriert.

7.2.1 Regularisierung mit Generative Adversarial Network

Für die Integration des generativen CNNs in ein GAN wird die Verlustfunktion des generativen CNNs durch die des GANs ergänzt, sodass die realen Spektren aus dem Trainingsdatensatz direkt in die Verlustfunktion einfließen (siehe Abbildung 7.3). Da die Verlustfunktion des generativen CNNs ergänzt wird, wird der Ansatz hier als regularisierendes *Generative Adversarial Network* (RGAN) bezeichnet. Durch die Verwendung des Diskriminators zusätzlich zum generativen CNN soll eine authentischere Modellierung der Spektren ermöglicht werden. Ein ähnlicher Ansatz existiert bereits für die unüberwachte spektrale Entmischung [52]. Im Allgemeinen wird die Regularisierung mit Hilfe des Diskriminators bereits für *Autoencoder* verwendet, wobei das Konzept als *Adversarial Autoencoder* bezeichnet wird [75]. Dort werden, anders als beim RGAN, die Werte der Engstelle regularisiert.

Im Gegensatz zum CGAN wird beim RGAN der Generator nicht nur indirekt über den Diskriminator trainiert, sondern auch direkt über die Spektren im Trainingsdatensatz. Damit soll der Zusammenhang zwi-

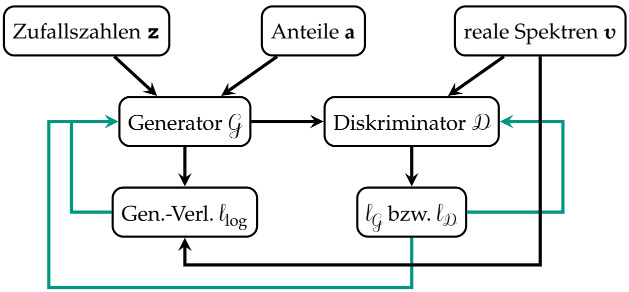


Abbildung 7.3 Schematische Darstellung des RGANs für die Erzeugung zusätzlicher Mischspektren [V10]. Schwarze Pfeile stehen für den Datenfluss in Vorwärtsrichtung, grüne für die Fehlerrückführung.

schen den Spektren und den Anteilsvektoren hergestellt werden. Weiterhin gibt es, wie in Abschnitt 7.1, die zufälligen Eingangsgrößen, damit für den gleichen Anteilsvektor mehrere Spektren realisiert werden können. Dadurch wird die Spektrenvariabilität modelliert.

Der Diskriminator wird wie beim ursprünglichen GAN trainiert. Sein Ziel besteht weiterhin darin, echte von generierten Spektren zu unterscheiden. Dabei versucht der Generator, möglichst authentische Daten zu generieren, sodass dem Diskriminator die Unterscheidung nicht mehr gelingt. Damit soll erreicht werden, dass die erzeugten Spektren so authentisch wie möglich sind.

Damit ist die Verlustfunktion des Diskriminators weiterhin (3.23), wobei die Spektren den Datenpunkten entsprechen. Die Verlustfunktion des Generators setzt sich aus (3.4) und (3.22) zusammen

$$l_{\mathrm{RGAN}} = l_{\mathcal{G}} + \mathrm{k_R} \cdot l_{\mathrm{log}}\,, \tag{7.5}$$

wobei die Spektren sowohl Eingangsdaten des Diskriminators als auch Ausgangsdaten des Generators sind und dementsprechend in die Formeln eingesetzt werden müssen. Dabei ist $\mathrm{k_R} \in \mathbb{R}^+$ ein Gewichtungsfaktor, der den Einfluss der beiden Teilverlustfunktionen steuert.

Nach dem Training kann der Generator analog zum generativen CNN aus Abschnitt 7.1 genutzt werden, um weitere Spektren für Anteilsvektoren zu erzeugen, die nicht im ursprünglichen Trainingsdatensatz enthalten sind. Auch hier wird die Spektrenvariabilität durch unterschiedliche

zufällige Eingangsgrößen realisiert. Der Diskriminator wird bei einem GAN normalerweise nicht für die Datenerzeugung herangezogen.

7.2.2 Diskriminator für Datenerzeugung

Bei dem hier vorgestellten Ansatz kann es durchaus Sinn ergeben, den Diskriminator bei der Datenerzeugung ebenfalls zu verwenden. Die Idee dahinter ist, dass hier Anteile vorgegeben werden, die der Generator nicht kennt. Somit besteht die Gefahr, dass er neue Spektren generiert, die weniger authentisch sind. Der Diskriminator wird jedoch dafür trainiert, um zu entscheiden, ob ein Spektrum echt erscheint oder nicht. Im Idealfall ist der Diskriminator auch in der Lage, dies für unbekannte Anteilsvektoren durchzuführen, da er mit verschiedenen Anteilsvektoren trainiert wurde. Wenn dies zutrifft, müsste durch die Verwendung eine Verbesserung der Ergebnisse erzielt werden können. Ob das der Fall ist, wird in Kapitel 8 untersucht.

Umgesetzt wird die Generation der Spektren so, dass die gewünschte Anzahl an Spektren pro Anteilsvektor vorgegeben wird. Dann werden für jeden Anteilsvektor so lange Spektren generiert, bis diese Zahl erreicht ist, wobei diejenigen, die der Diskriminator als unecht klassifiziert, direkt aussortiert werden. Wird für eine festgelegte Anzahl an erzeugten Spektren $\zeta_1 \in \mathbb{N}$ keines gefunden, das der Diskriminator für echt befindet, so wird die Entscheidungsgrenze, hier 0,5 (siehe Abschnitt 3.4), um einen festen Wert $\zeta_2 \in \mathbb{R}$ verkleinert. Diese Absenkung der Entscheidungsgrenze ist in der Praxis notwendig, da sonst für einige Anteilsvektoren die Spektren nie als echt klassifiziert werden, und wird für jeden vorgegebenen Anteilsvektor separat behandelt. Durch die schrittweise Absenkung werden trotzdem Spektren bevorzugt, bei denen der Diskriminator einen möglichst hohen Ausgangswert liefert, d. h. es werden Spektren bevorzugt, die nur knapp für unecht befunden wurden.

7.3 Gauß-Prozess-inspirierte neuronale Netze

Das in diesem Abschnitt vorgestellte Verfahren zur Augmentierung ist bereits veröffentlicht worden [V8], wobei in dieser Arbeit die Auswertung in Kapitel 8 ausführlicher ausfällt. Auch hier werden vor allem die

Augmentierung in der Albedo-Domäne und die spektrale Entmischung in Kapitel 8 zusätzlich betrachtet.

An dieser Stelle wird nicht im Detail auf stochastische Prozesse eingegangen, da dies für das Verständnis des Abschnitts nicht notwendig ist. Wichtig ist, dass ein stochastischer Prozess einer Zufallsvariablen entspricht, die von einer weiteren Eingangsgröße, meist der Zeit, abhängt. Damit ergibt sich für einen festen Zeitpunkt (eine feste Eingangsgröße) eine Zufallsvariable. Eine zeitabhängige Realisierung wird als Musterfunktion bezeichnet. In dieser Arbeit wird statt einer Zeitabhängigkeit die Abhängigkeit von der Wellenlänge genutzt. Daher wird auch die entsprechende Notation bei der Einführung verwendet.

Ein Gauß-Prozess $\mathrm{u}_\lambda \sim \mathcal{gp}\left(m_{\mathrm{u}}, k_{\mathrm{u}}\right)$ ist ein stochastischer Prozess, der vollständig durch seine Mittelwertfunktion $m_{\mathrm{u}} : \mathbb{R} \to \mathbb{R}$ und eine Kovarianzfunktion $k_{\mathrm{u}} : \mathbb{R}^2 \to \mathbb{R}$ beschrieben wird. Hier hängt der Gauß-Prozess, der die Spektren eines Datensatzes modelliert, vom Wellenlängenindex λ ab und ist zusätzlich mit dem Anteilsvektor $\mathbf{a}$ parametriert. Damit ist die Mittelwertfunkton

$$m_{\mathrm{u}}\left(\lambda|\mathbf{a}\right) \tag{7.6}$$

und die Kovarianzfunktion

$$k_{\mathrm{u}}\left(\lambda, \lambda'|\mathbf{a}\right) \tag{7.7}$$

mit den Wellenlängenindizes $\lambda, \lambda' \in \mathbb{N}^+$. Das Ziel besteht darin, diese beiden Zusammenhänge mit Hilfe vergleichsweise einfacher KNN zu modellieren. Dazu müssen zunächst die Daten, die als Spektren und Anteilsvektoren vorliegen, in eine bestimmte Form gebracht werden. Zur besseren Unterscheidung wird das KNN für die Mittelwertfunktion im Folgenden als $\mathcal{N}_{m_{\mathrm{u}}}$ und das für die Kovarianzfunktion als $\mathcal{N}_{k_{\mathrm{u}}}$ bezeichnet.

7.3.1 Vorbereitung der Trainingsdaten

Um die Trainingsdaten für $\mathcal{N}_{m_{\mathrm{u}}}$ und $\mathcal{N}_{k_{\mathrm{u}}}$ zu erhalten, müssen zunächst die Mittelwertvektoren und Kovarianzmatrizen der Spektren für jeweils einen Anteilsvektor berechnet werden. Diese werden aus den Spektrenmengen (7.1) mit Hilfe von (7.3) und (7.4) geschätzt. Die Elemente des Mittelwertvektors entsprechen (7.6) und die der Kovarianzmatrix (7.7),

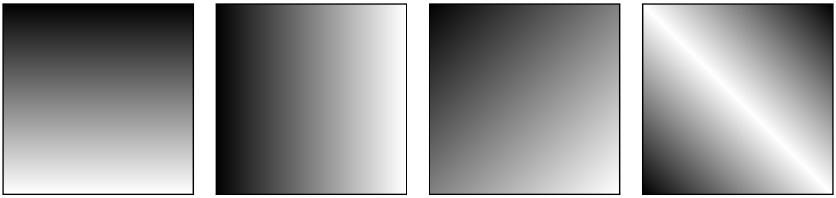


Abbildung 7.4 Illustration der Indizes $\lambda, \lambda', \tilde{\lambda}_1$ und $\tilde{\lambda}_2$ (von links nach rechts) innerhalb einer Kovarianzmatrix [V8]. Dabei steht Schwarz für kleine Werte und Weiß für große. Die schwarzen Rahmen dienen der besseren Darstellung und gehören nicht mehr zur eigentlichen Illustration.

jeweils für die im Trainingsdatensatz vorhandenen Wellenlängenindizes λ.

Die Eingangsdaten für das Training von $\mathcal{N}_{m_{\mathrm{U}}}$ sind nun der Anteilsvektor $\mathbf{a}$ und der Wellenlängenindex λ. Die Ausgangsdaten sind die korrespondierenden Elemente des Mittelwertvektors. Als Eingangsdaten für $\mathcal{N}_{k_{\mathrm{U}}}$ werden neben dem Anteilsvektor $\mathbf{a}$ die transformierten Wellenlängenindizes $\tilde{\lambda}_1 \in \mathbb{N}$ und $\tilde{\lambda}_2 \in \mathbb{N}$ mit

$$\tilde{\lambda}_1 = |\lambda + \lambda'| \tag{7.8}$$

und

$$\tilde{\lambda}_2 = \Lambda - |\lambda - \lambda'| \tag{7.9}$$

genutzt, wobei Λ der Anzahl der verwendeten Wellenlängenkanäle entspricht. Die Ausgangsdaten für das Training von $\mathcal{N}_{m_{\mathrm{U}}}$ sind die korrespondierenden Elemente der Kovarianzmatrix. Die Indizes $\tilde{\lambda}_1$ und $\tilde{\lambda}_2$ werden aufgrund zweier Eigenschaften von Kovarianzmatrizen benutzt. Zum einen sind sie symmetrisch und zum anderen sind die größten Werte auf der Hauptdiagonalen zu finden. Dies wird mit $\tilde{\lambda}_1$ und $\tilde{\lambda}_2$ nachempfunden (siehe Abbildung 7.4). Dank dieser Transformation muss das KNN weniger Änderungen in der Monotonie lernen und kann recht einfach gehalten werden. Im folgenden Abschnitt wird der Aufbau von $\mathcal{N}_{m_{\mathrm{U}}}$ und $\mathcal{N}_{k_{\mathrm{U}}}$ beschrieben, wobei die Dimensionen von Ein- und Ausgängen bereits durch die zuletzt beschriebenen Daten festgelegt sind, nämlich $P+1$ und 1 bzw. $P+2$ und 1.

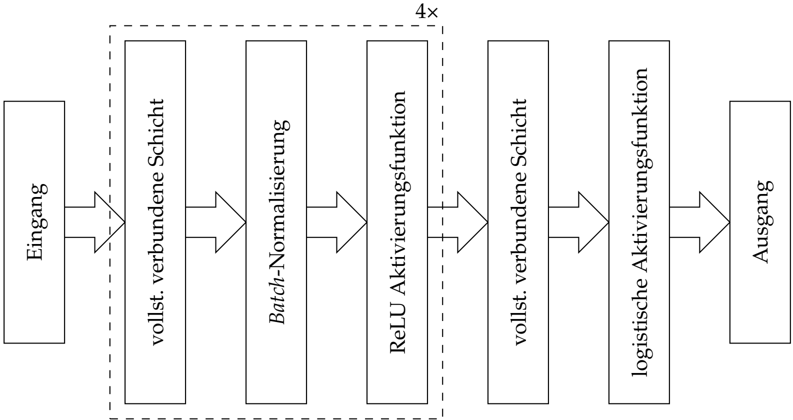


Abbildung 7.5 Schematische Darstellung des verwendeten neuronalen Netzwerks [V8]: Es hat vier Blöcke, die jeweils aus einer vollständig verbundenen Schicht, *Batch*-Normalisierung und der ReLU Aktivierungsfunktion bestehen. Am Ausgang befindet sich eine vollständig verbundene Schicht mit einer logistischen Aktivierungsfunktion.

7.3.2 Netzstruktur

Um die gewünschten Zusammenhänge (7.6) und (7.7) abbilden zu können, sind recht einfache KNN ausreichend. Wegen der kleinen Anzahl an Eingangsgrößen, die in keinerlei „Nachbarschaftsbeziehung" stehen (sie könnten in beliebiger Reihenfolge eingegeben werden), sind CNN hier nicht zielführend. Stattdessen werden vorwiegend vollständig verbundene Schichten verwendet. Der vollständige Aufbau beider KNN ist in Abbildung 7.5 zu sehen. Für das Training wird die logistische Verlustfunktion (3.4) genutzt. Diese hat während der Versuche die besten Ergebnisse erzielt, jedoch müssen die Werte der Ausgangsdaten zwischen 0 und 1 liegen. Während die Mittelwertvektoren durch die Normierung der Spektren schon Werte ausschließlich zwischen 0 und 1 besitzen, werden die Kovarianzmatrizen durch Verschiebung und Skalierung in diesen Bereich überführt. Am Ende der Datenerzeugung wird dies wieder rückgängig gemacht.

7.3.3 Augmentierungsstrategie

Die trainierten KNN können nun, wie bei den vorherigen Methoden, dazu genutzt werden, die dazugehörigen Mittelwertvektoren und Kovarianzmatrizen unter Vorgabe neuer Anteilsvektoren zu erzeugen. Da auch der Wellenlängenindex vorgegeben werden kann, wäre bei diesem Ansatz denkbar, Werte für Wellenlängen, die nicht im Trainingsdatensatz enthalten sind, zu erzeugen. Dies könnte sich vor allem bei der Verwendung unterschiedlicher Aufnahmeeinrichtungen mit unterschiedlicher spektraler Auflösung als nützlich erweisen, wurde jedoch im Kontext dieser Arbeit nicht untersucht. Dazu wäre es jedoch sinnvoller, statt den Wellenlängenindex direkt die Wellenlänge zu nutzen. Weil nur die Momente, nicht jedoch die Spektren selbst generiert werden, können die Spektren während des Trainings in jeder Epoche neu zufällig erzeugt werden, wie es auch bei der Methode in Abschnitt 6.2 der Fall ist.

Alle vorgestellten Verfahren zur Datenerzeugung (Kapitel 6) und zur Datenaugmentierung (Kapitel 7) werden nun in Kapitel 8 mit Hilfe von Experimenten mit mehreren Datensätzen ausgewertet.

8 Umsetzung und Analyse

In diesem Kapitel werden die in den Kapiteln 5 bis 7 vorgestellten Verfahren mit Hilfe von realen Datensätzen ausgewertet und verglichen. Dazu werden in Abschnitt 8.1 die verwendeten Datensätze und die Aufnahmeeinrichtung, mit der sie erstellt worden sind, vorgestellt. Anschließend werden in Abschnitt 8.2 die mit den entwickelten Augmentierungsverfahren erzeugten Spektren direkt mit Hilfe geeigneter Metriken bewertet. In Abschnitt 8.3 werden schließlich alle vorgestellten Verfahren bezüglich ihrer Performanz bei der spektralen Entmischung ausgewertet. Schließlich wird die Auswertung in Abschnitt 8.4 zusammengefasst.

8.1 Verwendete Datensätze

Bei den hier verwendeten Datensätzen handelt es sich um echte Aufnahmen aus dem Bildverarbeitungslabor des Instituts für Industrielle Informationstechnik am Karlsruher Institut für Technologie. Damit ist sichergestellt, dass die wahren Mischverhältnisse und damit die wahren Anteilsvektoren möglichst genau bekannt sind.

Die dort vorliegenden Bedingungen haben mit denen im industriellen Umfeld gemein, dass sowohl die Parameter der Aufnahmeeinrichtung als auch die Parameter der Beleuchtung bekannt sind bzw. bestimmt werden können. Des Weiteren besteht in beiden Fällen die Möglichkeit, einen Weißabgleich mit einem Reflexionsnormal durchzuführen (siehe Abschnitt 8.1.1). Auch treten viele der Ursachen für die Spektrenvariabilität (vgl. Abschnitt 2.5) in den Datensätzen auf. Ausnahmen sind hier die Effekte der Erdatmosphäre, die aber auch bei industriellen Anwendungen meist zu vernachlässigen sind.

Alle verwendeten Datensätze bestehen aus feinen Pulvern, sodass in jedem Pixel eine Mischung vorliegt, wenn es sich um eine Mischprobe handelt. Dadurch ist zu erwarten, dass sich die Mischungen wie in Ab-

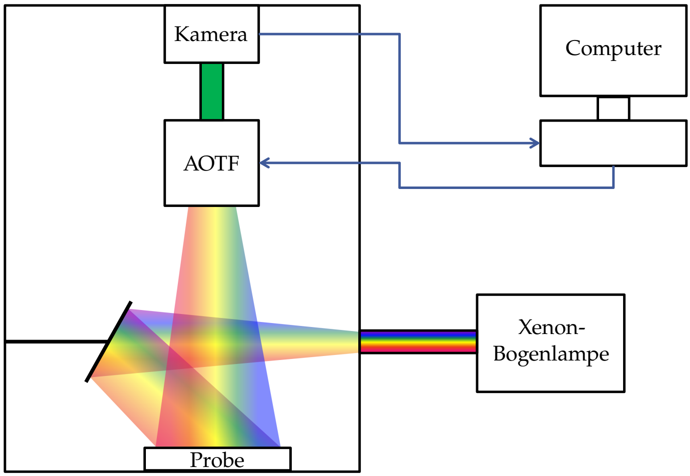


Abbildung 8.1 Schematische Darstellung des Messaufbaus.

schnitt 2.4.2 verhalten. Damit besteht zwischen den Reinspektren und den Spektren der Stoffgemische ein nichtlinearer Zusammenhang. Dies ist bei der Auswertung der auf KNN basierenden Ansätze besonders interessant, da die Stärke von KNN darin besteht, beliebige Funktionen approximieren zu können.

8.1.1 Messaufbau

Der Messaufbau, der zur Aufnahme der hyperspektralen Bilder bzw. Spektren verwendet worden ist, wird in Abbildung 8.1 dargestellt [5, 67]. Die Aufnahmen werden in einer Dunkelkammer durchgeführt, um Störeinflüsse durch Fremdlicht möglichst ausschließen zu können. Als Lichtquelle dient eine 300 W Xenon-Bogenlampe des Herstellers LOT-QuantumDesign. Diese liefert über den gesamten relevanten Wellenlängenbereich Licht mit ausreichender Leistung[1]. Das Licht wird über einen Spiegel auf die zu messende Probe gelenkt.

[1] Datenblatt: www.qd-europe.com/fileadmin/Mediapool/products/lightsources/pdf/Lamp_spectra_and_irradiance.pdf



Abbildung 8.2 Setzkästen mit Quarzsandmischungen links und Farbpulvermischungen rechts [V7, V10].

Das von der Messprobe gestreute Licht trifft durch ein *acousto-optic tunable filter* (AOTF) auf eine EMCCD-Kamera (*electron multiplying charge-coupled device*). Mit Hilfe des AOTFs können Wellenlängenbereiche ausgewählt werden, die durchgelassen werden. Alle anderen werden gesperrt. So können für die Wellenlängenbereiche hintereinander Grauwertbilder aufgenommen werden, die anschließend zu einem hyperspektralen Bild zusammengesetzt werden. Als AOTF kommt das Modell HSi-300 des Herstellers Gooch&Housego zum Einsatz, als Kamera das Modell iXon$_3$897 EMCCD des Herstellers Andor. Ein Computer steuert beide Geräte und speichert die Bilder ab.

Die hyperspektrale Messeinrichtung deckt einen Wellenlängenbereich von circa 448 nm bis circa 812 nm ab, der für die Aufnahme der Datensätze voll ausgeschöpft wird. Die Bandbreiten der einzelnen Wellenlängenkanäle variieren leicht, da sich beim AOTF prinzipbedingt nicht alle Bandbreiten für jede mittlere Wellenlänge einstellen lassen. Sie werden so gewählt, dass sie möglichst nahe an 4 nm sind. Damit lässt sich der Wellenlängenbereich in $\Lambda = 91$ Wellenlängenkanäle einteilen, wobei die mittleren Wellenlängen der Kanäle von 450 nm in 4 nm-Schritten bis 810 nm reichen.

Zur Erstellung der Mischproben sind die Pulver entsprechend ihrer vorgegebenen Volumenanteile bzw. Anteilsvektoren (siehe Kapitel 2) in Dosen gefüllt worden. Durch Schütteln sind anschließend homogene Pulvermischungen entstanden. Sowohl die Mischproben als auch die reinen Pulver sind zur Aufnahme in Setzkästen gefüllt worden (siehe Abbildung 8.2). Für die Datensätze werden von jeder Probe 20×20 Pixel große

Ausschnitte verwendet, sodass 400 Spektren für jedes Mischverhältnis zur Verfügung stehen. Damit wird sichergestellt, dass keine Ränder der Setzkästen oder deren Schatten im Datensatz eine Rolle spielen. Dies gilt nicht für Schatten, die die Proben selbst werfen, wie es bspw. rechts in Abbildung 8.2 der Fall ist.

Als Referenz ist unter gleichen Bedingungen eine Aufnahme mit einem Reflexionsnormal des Herstellers Spectralon angefertigt worden. Durch eine elementweise Division der Aufnahmen der Proben und der Referenzaufnahme wird die relative Reflektanz (2.4) berechnet, was auch als Weißabgleich bezeichnet wird. Die daraus resultierenden Spektren werden als Datenpunkte für die Auswertung der vorgestellten Verfahren verwendet.

8.1.2 Datensätze bestehend aus gefärbten Quarzsanden

Zur Erstellung der Datensätze sind gefärbte Quarzsande des Herstellers Qsand[2] verwendet worden. Für den ersten Datensatz, der von hier an als $\mathcal{X}_{\mathrm{Q3}}$ bezeichnet wird, sind die Quarzsande mit den Bezeichnungen „Blau 5/31“, „Rot 3/26“ und „Gelb 1/17“ als Reinstoffe verwendet worden [66]. Die mittleren Spektren der Reinstoffe sind in Abbildung 8.3 zu sehen, wobei die gefärbten Flächen die Spektrenvariabilität andeuten.

Die reinen Quarzsande sind systematisch, basierend auf unterschiedlichen Anteilsvektoren, miteinander vermischt worden. Dies ist so geschehen, dass alle möglichen Kombinationen an Anteilen erstellt worden sind, wobei die Schrittweite zwischen den Anteilen, die Anteilsstufe, $\mathcal{S} = \frac{1}{8}$ beträgt. Die daraus resultierenden 45 Anteilsvektoren sind in Abbildung 8.4 illustriert.

Ein zweiter Datensatz, der als $\mathcal{X}_{\mathrm{Q4}}$ bezeichnet wird, bestehend aus gefärbten Quarzsanden desselben Herstellers, verwendet als Reinstoffe die Quarzsande mit den Bezeichnungen „Grün 6/18“, „Rot 3/4“, „Blau 5/104“ und „Gelb 1/59“ [66]. Die mittleren Spektren der Reinstoffe sind in Abbildung 8.5 zu sehen, wobei die gefärbten Flächen die Spektrenvariabilität andeuten.

[2] http://www.qsand.eu

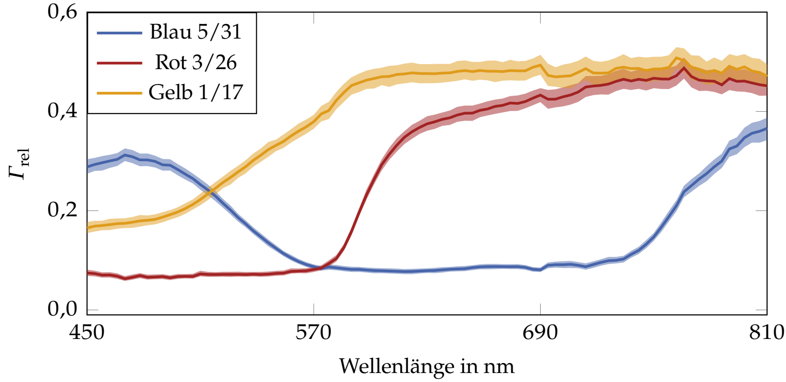


Abbildung 8.3 Reinspektren des Datensatzes $\mathcal{X}_{Q3}$. Die Linien entsprechen den mittleren Spektren, während die gefärbten Flächen die Spektrenvariabilität darstellen. Um den Einfluss von Ausreißern zu verringern, stellt die untere Grenze der Flächen das 5. Perzentil und die obere das 95. Perzentil dar (Bestimmung unabhängig für jeden Wellenlängenkanal).

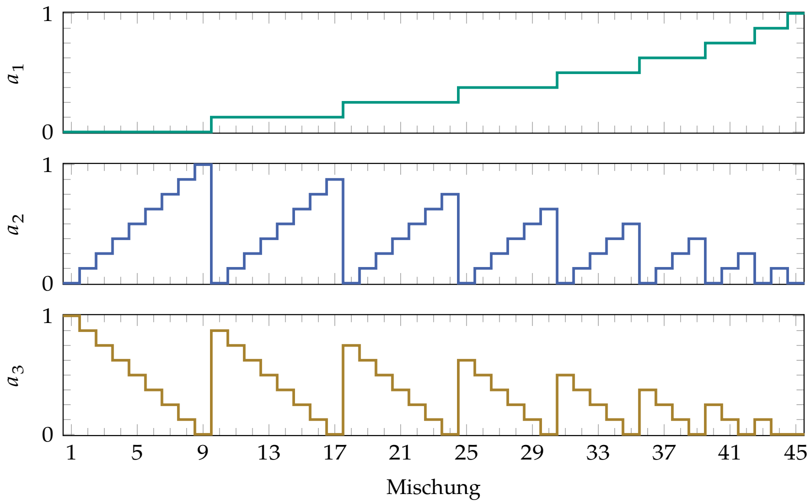


Abbildung 8.4 Resultierende Anteilsvektoren bei drei Komponenten und $\mathcal{S} = \frac{1}{8}$. Die Mischungen bzw. Anteilsvektoren sind durchnummeriert, wobei die Anteile der Komponenten separat dargestellt sind.

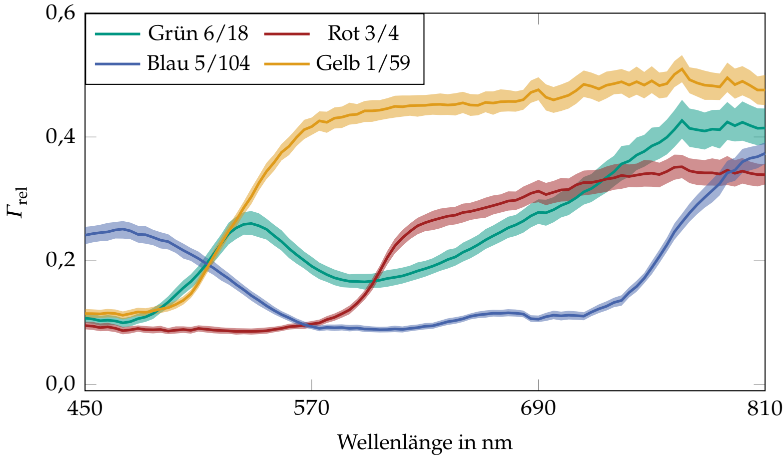


Abbildung 8.5 Reinspektren des Datensatzes $\mathcal{X}_{Q4}$. Die Linien entsprechen den mittleren Spektren, während die gefärbten Flächen die Spektrenvariabilität darstellen. Um den Einfluss von Ausreißern zu verringern, stellt die untere Grenze der Flächen das 5. Perzentil und die obere das 95. Perzentil dar (Bestimmung unabhängig für jeden Wellenlängenkanal).

Hier wird als Anteilsstufe $\mathcal{S} = \frac{1}{5}$ verwendet. Damit sind hier mehr Komponenten enthalten, jedoch besteht ein größerer Abstand zwischen den resultierenden 56 Anteilsvektoren, die in Abbildung 8.6 dargestellt sind. Aus den dargestellten Anteilsvektoren wird deutlich, wie aufwendig es ist, Datensätze zu erstellen, wenn die Zahl der Komponenten steigt und die Anteilsstufe kleiner wird. Daher ist die Verwendung von Verfahren sinnvoll, die die Trainingsdaten künstlich erweitern.

Die für die Datensätze verwendeten Quarzsande zeichnen sich dadurch aus, dass die einzelnen Körner nahezu sphärisch sind und der Durchmesser nur schwach variiert (circa 100 nm bis 300 nm). Dadurch sind die Vereinfachungen, die in den Abschnitten 2.4.2 und 2.6.2 getroffen werden, durchaus gerechtfertigt. Durch die kleine Korngröße ist zu erwarten, dass sich die Mischungen wie Mischungen im mikroskopischen Maßstab nach Abschnitt 2.4.2 verhalten. Damit sollten die daraus resultierenden Entmischungsverfahren innerhalb der modellbasierten Verfahren zu den besten Ergebnissen führen. Beispiele für Mischungen in Setzkästen sind in Abbildung 8.2 (links) dargestellt.

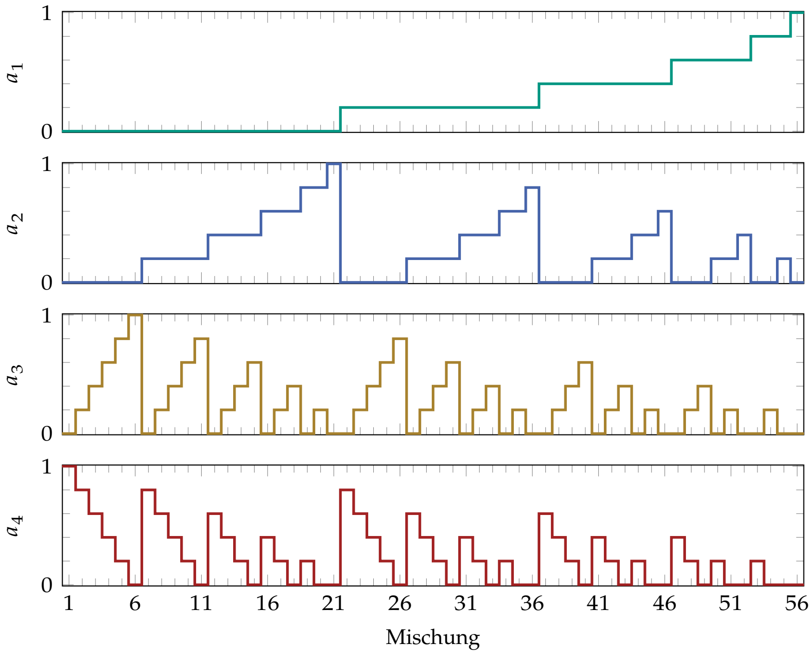


Abbildung 8.6 Resultierende Anteilsvektoren bei drei Komponenten und $\mathcal{S} = \frac{1}{5}$. Die Mischungen bzw. Anteilsvektoren sind durchnummeriert, wobei die Anteile der Komponenten separat dargestellt sind.

8.1.3 Farbpulverdatensatz

Der dritte Datensatz besteht aus Farbpigmenten des Herstellers Kremer Pigmente GmbH & Co. KG[3]. Für den hier verwendeten Datensatz $\mathcal{X}_{F4}$ sind die Farbpulver mit den Bezeichnungen „Chromoxid-Grün“, „Eisenglimmer-Violett“, „Ultramarin-Blau“ und „Eisenoxid-Gelb“ genutzt worden [5]. Die mittleren Spektren der Reinstoffe sind in Abbildung 8.7 zu sehen, wobei die gefärbten Flächen die Spektrenvariabilität andeuten. Auch beim Datensatz $\mathcal{X}_{F4}$ wird die Anteilsstufe $\mathcal{S} = \frac{1}{5}$ genutzt.

[3] www.kremer-pigmente.com

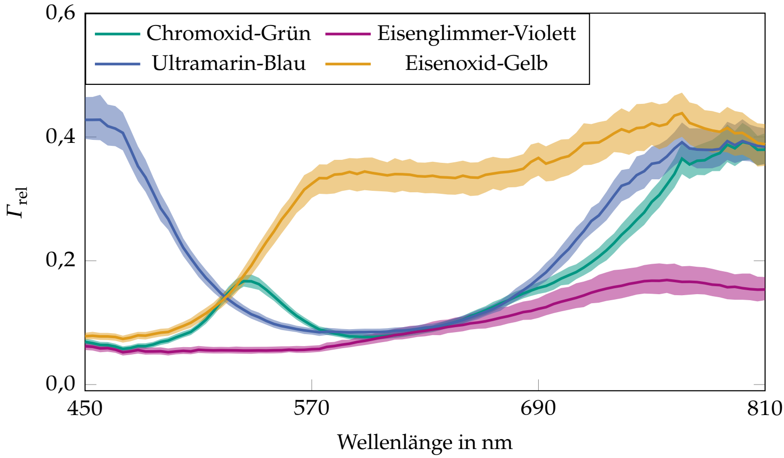


Abbildung 8.7 Reinspektren des Datensatzes $\mathcal{X}_{F4}$. Die Linien entsprechen den mittleren Spektren, während die gefärbten Flächen die Spektrenvariabilität darstellen. Um den Einfluss von Ausreißern zu verringern, stellt die untere Grenze der Flächen das 5. Perzentil und die obere das 95. Perzentil dar (Bestimmung unabhängig für jeden Wellenlängenkanal).

Im Gegensatz zu den Quarzsanden weisen die Partikel der Farbpulver eine hohe Varianz in ihrer Größe auf. Die Durchmesser reichen von wenigen Nanometern bis zu 600 nm. Damit ist eine Modellierung nach Abschnitt 2.4.2 ungenauer. Darüber hinaus neigen die Farbpulver zur Klümpchenbildung, was zu einer unregelmäßigen Oberfläche samt Schattenwurf führt (siehe Abbildung 8.2, rechts). Dies führt zu einer höheren Spektrenvariabilität, was auch in Abbildung 8.7 zu sehen ist. Insgesamt ist daher zu erwarten, dass die spektrale Entmischung des Datensatzes $\mathcal{X}_{F4}$ am anspruchsvollsten ist, verglichen mit den Datensätzen $\mathcal{X}_{Q3}$ und $\mathcal{X}_{Q4}$.

8.1.4 Datensätze in der Albedo-Domäne

Für viele der folgenden Ergebnisse findet ein Vergleich zwischen den Ergebnissen bei Nutzung der vorgestellten Datensätze, die aus Reflektanzspektren bestehen, und der in die Albedo-Domäne transformierten Daten statt. Die Transformation erfolgt mit (2.11), wobei die Winkel $\beta_0 = 30°$ und $\beta = 0°$ genutzt werden. Das entspricht den Winkeln von Ka-

mera und Beleuchtung am Mittelpunkt des Setzkastens. Abweichungen nach Außen werden vernachlässigt. Die Oberflächennormale ist ohnehin nicht bekannt und darüber hinaus sind die Unterschiede in Abhängkeit der Winkel klein (siehe Abbildung 2.4). Der dadurch entstehende Fehler ist in Anbetracht der erreichbaren Genauigkeit vernachlässigbar (vgl. Ergebnisse in Abschnitt 8.3).

8.1.5 Aufteilung der Datensätze

Für die Verwendung und Beurteilung maschineller Lernverfahren müssen die Datensätze jeweils in einen Trainings-, einen Validierungs- und einen Testteil aufgeteilt werden (siehe Abschnitt 3.1.3).

Der Testteil ist bei allen verwendeten Methoden gleich und wird auch bei Verfahren genutzt, die kein Training benötigen, um eine Vergleichbarkeit der Resultate gewährleisten zu können. Da die hier vorgestellten Verfahren dahingehend bewertet werden sollen, wie gut sie die Mischeigenschaften der Datensätze abbilden, ist eine zufällige Einteilung der Datenpunkte nicht zielführend. Stattdessen ist die Trennung nach Anteilsvektoren sinnvoll, sodass in den Testdatensätzen nur unbekannte Anteilsvektoren enthalten sind. Andernfalls besteht die Gefahr, dass, wie bei einer Klassifikation, die bekannten Anteilsvektoren gelernt werden, aber keine Zwischenwerte, die für die Generalisierung bei der spektralen Entmischung benötigt werden. Um zusätzlich sogar einzelne Anteilswerte exklusiv im Testdatensatz zu haben, erfolgt die Aufteilung systematisch.

Beim Datensatz $\mathcal{X}_{\mathrm{Q3}}$ mit 3 Komponenten und einer Anteilsstufe von $\mathcal{S} = \frac{1}{8}$ werden alle Spektren für den Testdatensatz genutzt, bei denen mindestens ein Anteilswert nicht ganzzahlig durch $\frac{1}{4}$ teilbar ist. Das ist für insgesamt 30 Anteilsvektoren der Fall. Bei den Datensätzen $\mathcal{X}_{\mathrm{Q4}}$ und $\mathcal{X}_{\mathrm{F4}}$ kommen die Anteilsvektoren (und die dazugehörigen Spektren) in den Testdatensatz, bei denen mindestens ein Anteil den Wert $\frac{1}{5}$ oder den Wert $\frac{4}{5}$ hat. Das ist für insgesamt 40 Anteilsvektoren der Fall.

Im Gegensatz zu den Testdaten, die für alle Methoden gleich sind, variieren die Trainings- und die Validierungsdaten in Abhängigkeit der verwendeten Methode. Generell gilt, dass von den Daten, die für das Training bereitstehen, zufällige 10 % für die Validierung aufgespart werden. Für die Verfahren aus Kapitel 6 gilt:

- Die Trainings- und Validierungsdaten für das CNN zur spektralen Entmischung werden im Verhältnis von 9 zu 1 modellbasiert aus den Reinspektren erzeugt.
- Die verbleibenden Spektren (also ohne Reinspektren und Testdaten) werden beim GBM und beim LQM als Validierungsdaten zur Bestimmung der Nichtlinearitätskoeffizienten genutzt.

Für die Verfahren aus Kapitel 7 gilt:

- Für die KNN zur Datenerzeugung werden die Daten, die nicht Testdaten sind, zufällig in 90 % Trainings- und 10 % Validierungsdaten eingeteilt.
- Für die CNN zur spektralen Entmischung werden die bereits augmentierten Daten zufällig in 90 % Trainings- und 10 % Validierungsdaten eingeteilt.

Mit diesen Datensätzen werden nun in den folgenden Abschnitten die vorgestellten Methoden bewertet. Dabei ist allen Datensätzen gemein, dass für jeden Anteilsvektor viele Spektren existieren. Dies ist im industriellen Umfeld plausibel (vgl. Abschnitt 1.2), da die Erstellung von Stoffgemischen meist wesentlich aufwendiger ist als die Aufnahme unterschiedlicher Spektren derselben Gemische.

8.2 Erzeugte Spektren

In diesem Abschnitt werden die KNN aus Kapitel 7, die Daten für neue Anteilsvektoren erzeugen und damit vorhandene Datensätze vergrößern, bewertet. Hier werden zunächst die erzeugten Spektren direkt bewertet, indem sie mit Spektren aus den Testdatensätzen verglichen werden. In Abschnitt 8.3 folgt eine Bewertung anhand Leistung bei der spektralen Entmischung.

8.2.1 Gütemaße

Damit die erzeugten Spektren mit den Spektren der Testdatensätze verglichen werden können, werden nach dem Training der KNN die Anteilsvektoren zur Datenerzeugung vorgegeben, die auch in den Testdatensätzen

zu finden sind. Da in allen Verfahren aus Kapitel 7 die Spektrenvariabilität über eine zufällige Komponente berücksichtigt wird, ist eine Eins-zu-eins-Zuordnung von erzeugten und echten Spektren nicht möglich. Daher werden immer Mengen von Spektren miteinander verglichen, die mit dem gleichen Anteilsvektor korrespondieren. In den Testdatensätzen befinden sich jeweils 400 Spektren in jeder Menge. Bei der Datenerzeugung können, je nach Verfahren, sehr viele unterschiedliche Spektren erzeugt werden, jedoch werden zur Gewinnung der folgenden Ergebnisse ebenfalls 400 Spektren für jeden Anteilsvektor erzeugt.

Ein Vergleichskriterium solcher Spektrenmengen ist bisher nicht bekannt. Naheliegend sind Maße, welche die Unterschiede zwischen Wahrscheinlichkeitsverteilungen ermitteln, wie bspw. die Kullback-Leibler-Divergenz [69]. Diese Maße scheitern jedoch in der Praxis an der hohen Dimension der Spektren. Bei den vorhandenen Daten würden 91-dimensionale Wahrscheinlichkeitsdichtefunktionen miteinander verglichen werden. Bei jeweils 400 Datenpunkten ist es sehr unwahrscheinlich, dass überhaupt exakt gleiche Spektren in beiden Spektrenmengen vorhanden sind. In einem Vergleich haben damit die (aus den Daten geschätzten) Wahrscheinlichkeitsdichten an nahezu allen Stellen den Wert 0.

Daher werden für die Bewertung der Spektrenmengen zwei Gütemaße eingeführt, die auf dem spektralen Winkel (engl. *spectral angle*, SA)

$$\epsilon_{\mathrm{SA}} = \arccos\left(\frac{\langle \hat{\boldsymbol{v}}, \boldsymbol{v}^* \rangle}{\|\hat{\boldsymbol{v}}\| \cdot \|\boldsymbol{v}^*\|}\right) \tag{8.1}$$

basieren, der auf der Kosinus-Ähnlichkeit zwischen den Spektren $\hat{\boldsymbol{v}}$ und $\boldsymbol{v}^*$ basiert. Er ist ein gängiges Gütekriterium bei der Bewertung der Ähnlichkeit von Spektren.

Um die Lage der Spektrenmengen insgesamt zu bewerten, wird als erstes Gütemaß der spektrale Winkel zwischen den mittleren Spektren (engl. *mean spectra spectral angle*, MSSA) der erzeugten Spektrenmenge $\hat{Y}_{\mathbf{a}}$ und der wahren Spektrenmenge $Y^*_{\mathbf{a}}$ eingeführt:

$$\epsilon_{\mathrm{MSSA}} = \arccos\left(\frac{\left\langle \hat{\boldsymbol{\mu}}_{\hat{Y}_{\mathbf{a}}}, \hat{\boldsymbol{\mu}}_{Y^*_{\mathbf{a}}} \right\rangle}{\left\|\hat{\boldsymbol{\mu}}_{\hat{Y}_{\mathbf{a}}}\right\| \cdot \left\|\hat{\boldsymbol{\mu}}_{Y^*_{\mathbf{a}}}\right\|}\right). \tag{8.2}$$

Bei den Mittelwerten handelt es sich um Stichprobenmittelwerte gemäß (7.3). Als zweites eingeführtes Gütemaß wird der durchschnittliche minimale spektrale Winkel (engl. *average minimal spectral angle*, AMSA) eingeführt:

$$\epsilon_{\mathrm{AMSA}} = \frac{1}{S}\sum_{s=1}^{S}\min_{\tilde{s}}\left(\arccos\left(\frac{\left\langle \hat{\boldsymbol{v}}_{\mathbf{a}\tilde{s}}, \boldsymbol{v}^*_{\mathbf{a}s}\right\rangle}{\left\|\hat{\boldsymbol{v}}_{\mathbf{a}\tilde{s}}\right\| \cdot \left\|\boldsymbol{v}^*_{\mathbf{a}s}\right\|}\right)\right). \tag{8.3}$$

Hier fließt für alle wahren Spektren $\boldsymbol{v}^*_{\mathbf{a}s}$ einer Spektrenmenge im Testdatensatz das erzeugte Spektrum $\hat{\boldsymbol{v}}_{\mathbf{a}\tilde{s}}$ ein, welches den kleinsten spektralen Winkel zu ersterem aufweist. Um die ganze Spektrenmenge abzubilden, wird davon der Mittelwert gebildet. Das Minimum wird wegen der erwähnten fehlenden Zuordenbarkeit der Spektren genutzt. So wird geprüft, ob es zu jedem Spektrum ein möglichst ähnliches erzeugtes Spektrum gibt, bzw., wie weit die Spektren auseinander liegen. Mit dem AMSA soll vor allem die Spektrenvariabilität der erzeugten Spektren bewertet werden, mit dem MSSA die Mischcharakteristik. Letztere wird jedoch ebenfalls durch den AMSA bewertet, sodass am besten immer beide Gütemaße betrachtet werden.

Alle KNN zur Erzeugung von Augmentierungsdaten werden auch mit Spektren in der Albedo-Domäne (siehe Abschnitt 2.4.2) trainiert. Auf diese Weise soll geprüft werden, ob damit bessere Ergebnisse erzielt werden können. Es handelt sich nämlich bei allen untersuchten Datensätzen um Pulvermischungen, die sich in der Albedo-Domäne einfacher modellieren lassen (siehe Abschnitt 2.6.2). Die vorgestellten Gütemaße werden hier immer auf die Reflektanzspektren angewandt, wie sie auch in den Testdatensätzen enthalten sind. Werden Spektren durch die Verfahren in der Albedo-Domäne erzeugt, werden diese vor Berechnung der Gütemaße mit (2.10) in die Reflektanz-Domäne transformiert. Würden stattdessen die Testspektren zur Berechnung der Gütemaße in die Albedo-Domäne transformiert, so würden sich die spektralen Winkel zwischen den Spektren und somit die verwendeten Gütemaße ändern.

8.2.2 Konfiguration der Verfahren

Während in Kapitel 7 die Methoden an sich beschrieben sind, werden in diesem Abschnitt alle Parameter dargestellt, die bei der Erzeugung der

Ergebnisse genutzt werden. Bestimmt werden die Parameter durch eine manuelle Rastersuche mit Hilfe der Validierungsdatensätze.

8.2.2.1 Generatives Faltungsnetz

Die Struktur des generativen Faltungsnetzes (engl. *generative convolutional neural network*, GCNN) ist bereits in Abbildung 7.1 dargestellt. Für die folgende Auswertung werden in jeder Schicht Faltungskerne der Länge $I = 5$ verwendet. In der ersten Schicht werden abweichend davon Faltungskerne der Länge $I = 23$ genutzt. Das Auffüllen mit Nullen in den transponierten Faltungsschichten erfolgt mit einem Faktor von 2, was am Ausgang 92 Werte ergibt. Da davon nur 91 für die zur Verfügung stehenden Datensätze benötigt werden, wird der letzte Ausgangswert ignoriert. Die Anzahlen der Merkmalskarten sind, vom Eingang zum Ausgang, in der jeweiligen Schicht $P + Z$, 32, 32, 16 und 1. Dabei sind P die Anzahl der Komponenten im Datensatz, hier also 3 bzw. 4, und Z die Anzahl der zufälligen Eingangsgrößen. Für letztere hat sich der Wert $Z = 3$ als sinnvoll erwiesen. Für jede der zufälligen Eingangsgrößen wird eine Standardnormalverteilung verwendet. Es hat sich herausgestellt, dass die Verwendung einer Gleichverteilung keinen Unterschied macht, da sich das GCNN an die verwendete Verteilung anpasst.

Beim Training wird der Adam-Optimierer (siehe Abschnitt 3.2.3.2) mit einer Lernrate von $\alpha = 0{,}01$ verwendet. Für die restlichen Parameter des Adam-Optimierers werden $\beta_1 = 0{,}9$ und $\beta_2 = 0{,}999$ verwendet (wie von Kingma und Ba [61] empfohlen). Das Training wird für jeweils 4000 Epochen durchgeführt, wobei in jedem Trainingsschritt der vollständige Trainingsdatensatz verwendet wird. Trotz der großen Zahl an Epochen bereitet Überanpassung hier keine Probleme, was die Überlegungen in Abschnitt 7.1 unterstreicht.

Ebenfalls in die Ergebnisse fließen die Varianten des GCNNs ein. Zum einen wird, wie in Abschnitt 7.1.3.1 beschrieben, die Varianz der zufälligen Eingänge bei der Datenerzeugung verdoppelt. Diese Variante wird in den Ergebnissen mit GCNN-2 gekennzeichnet. Zum anderen wird die Regularisierung mit der Kovarianzmatrix aus Abschnitt 7.1.3.2 betrachtet, was mit GCNN-Cov bezeichnet wird. Bei letzterer wird für k_{Σ} aus (7.2) der Wert 10^7 genutzt. Dieser große Wert resultiert aus den sich stark unterscheidenden Wertebereichen der Teilverlustfunktionen.

8.2.2.2 Regularisierendes Generative Adversarial Network

Bei der Realisierung des RGANs entspricht der Generator dem im vorhergehenden Abschnitt vorgestellten GCNN. Als Diskriminator wird ein KNN verwendet, das aus drei vollständig verbundenen Schichten besteht. Die Anzahl an Neuronen beträgt vom Eingang zum Ausgang 64, 16, und 1. Als Aktivierungsfuntkion wird in jeder Schicht des Diskriminators die logistische Funktion (3.7) verwendet, welche sich in den Versuchen als vorteilhaft herausgestellt hat.

Darüber hinaus hat sich gezeigt, dass es zu besseren Ergebnissen führt, wenn die Gewichte von Generator und Diskriminator nicht alternierend angepasst werden. Stattdessen werden die des Generators nur in jedem siebten Trainingsschritt trainiert, wobei in der Verlustfunktion (7.5) $k_R = 10$ genutzt wird. Bei allen anderen Trainingsschritten wird nur der Diskriminator trainiert.

Beim Training wird ebenfalls der Adam-Optimierer mit $\beta_1 = 0{,}9$ und $\beta_2 = 0{,}999$ verwendet. Außerdem wird in jedem Trainingsschritt der vollständige Trainingsdatensatz genutzt, sodass die Anzahl an Epochen der Anzahl der Trainingsschritte entspricht. Für jeden Datensatz werden eine individuelle Anzahl an Epochen und eine individuelle Lernrate verwendet. Der Datensatz $\mathcal{X}_{Q3}$ wird mit einer Lernrate $\alpha = 0{,}01$ für 6000 Epochen trainiert. Bei den Datensätzen $\mathcal{X}_{Q4}$ und $\mathcal{X}_{F4}$ wird ein Vortraining durchgeführt, bei dem nur der Generator mit der Verlustfunktion (3.4) trainiert wird. Dieses Vortraining erfolgt bei beiden Datensätzen für 450 Epochen mit $\alpha = 0{,}01$. Anschließend werden beide Datensätze mit $\alpha = 10^{-3}$ für 4000 Epochen trainiert. Das Vortraining ist bei den Datensätzen mit 4 Komponenten notwendig, da hier die Anteilsvektoren weiter auseinander liegen als bei Datensatz $\mathcal{X}_{Q3}$. Dadurch ist der Mischzusammenhang für das RGAN schwieriger zu erfassen.

Wird bei der Datenerzeugung ebenfalls der Diskriminator verwendet, was hier mit RGAN-DE abgekürzt wird, werden die Parameter $\zeta_1 = 100$ und $\zeta_2 = 0{,}01$ verwendet (siehe Abschnitt 7.2.2).

8.2.2.3 Gauß-Prozess-inspirierte neuronale Netze

Bei der Augmentierung mit Gauß-Prozess-inspirierten neuronalen Netzen (GPNN) werden die Netze wie in Abbildung 7.5 realisiert, wobei in jeder vollständig verbundenen Schicht 32 Neuronen verwendet werden.

Für das Training der neuronalen Netze $\mathcal{N}_{m_{\mathrm{U}}}$ und $\mathcal{N}_{k_{\mathrm{U}}}$ wird bei allen Datensätzen der Adam-Optimierer mit den Parametern $\alpha = 0{,}01$, $\beta_1 = 0{,}9$ und $\beta_2 = 0{,}999$ verwendet. Die Anzahl an Epochen ist unterschiedlich. Bei den Datensätzen $\mathcal{X}_{\mathrm{Q3}}$ und $\mathcal{X}_{\mathrm{Q4}}$ werden beide neuronale Netze für 2000 Epochen trainiert. Beim Datensatz $\mathcal{X}_{\mathrm{F4}}$ wird das neuronale Netz $\mathcal{N}_{m_{\mathrm{U}}}$ für 3000 Epochen trainiert, während $\mathcal{N}_{k_{\mathrm{U}}}$ für 4000 Epochen trainiert wird. Auch hier hat es sich als vorteilhaft erwiesen, den ganzen Trainingsdatensatz in jedem Trainingsschritt zu verwenden.

Als Vergleich wird in den Ergebnissen eine Variante dieses Verfahrens untersucht, bei dem nur die Mittelwertvektoren zur Augmentierung genutzt werden. Diese wird mit GPNN-M abgekürzt. So kann der Einfluss der Kovarianzfunktion ermittelt werden.

8.2.3 Ergebnisse

Für die Ergebnisse wird für jeden Anteilsvektor im Testdatensatz eine Menge an Spektren mit den vorgestellten Verfahren erzeugt. Für jeden Anteilsvektor werden damit und mit den dazugehörigen Testspektren die Gütemaße ϵ_{MSSA} und ϵ_{AMSA} berechnet. Der Prozess aus Training und Datenerzeugung wird jeweils 5 mal durchgeführt. Für jeden Anteilsvektor wird der Mittelwert von ϵ_{MSSA} und ϵ_{AMSA} bezüglich der Wiederholungen berechnet und in den Ergebnissen dargestellt. Da es sich bei allen Ergebnissen in diesem Abschnitt um Mittelwerte bezüglich der Wiederholungen handelt, werden diese nicht extra gekennzeichnet.

Die für die Darstellung verwendeten Box-Plots deuten also die Verteilung der Gütemaße bezüglich der unterschiedlichen Anteilsvektoren an. Bei den Box-Plots in diesem Abschnitt schließen die „Antennen“ alle Anteilsvektoren eines Datensatzes ein. Die Box geht vom 25. bis zum 75. Perzentil und der Strich gibt den Median an. Alle Verfahren werden sowohl auf Reflektanzspektren als auch auf Spektren, die zuvor in die Albedo-Domäne transformiert wurden, angewandt. Zur Berechnung

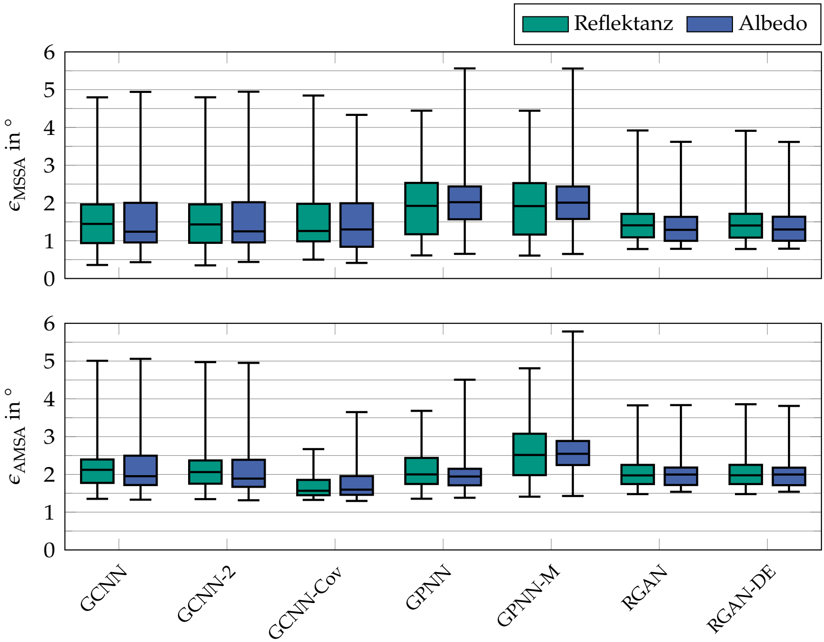


Abbildung 8.8 Darstellung von ϵ_{MSSA} und ϵ_{AMSA} als Box-Plots für den Datensatz $\mathcal{X}_{\text{Q3}}$.

von ϵ_{MSSA} und ϵ_{AMSA} werden letztere zuerst rücktransformiert (siehe Abschnitt 8.2.1).

In Abbildung 8.8 ist oben das Gütemaß ϵ_{MSSA} für den Datensatz $\mathcal{X}_{\text{Q3}}$ zu sehen. Den besten Median weist das Verfahren GCNN-Cov auf. Das lässt vermuten, dass sich die Regularisierung mit den Kovarianzmatrizen positiv auf das mittlere Spektrum einer Spektrenmenge auswirkt. Eine bessere Streuung erreicht das RGAN bei ähnlichem Median. Auch hier scheint die Regularisierung positive Auswirkungen zu haben. Den schlechtesten Median weisen die Verfahren GPNN und GPNN-M auf. Dabei sind die neuronalen Netze jedoch kleiner als bei den übrigen Verfahren. Der Einsatz der Verfahren in der Albedo-Domäne liefert ähnliche Ergebnisse. Sie sind in Abhängigkeit des genutzten Verfahrens zur Datenerzeugung teilweise leicht besser, teilweise leicht schlechter. Hier lernen die neuronalen Netze die nichtlinearen Zusammenhänge bei Ver-

wendung der Reflektanzspektren ähnlich gut wie bei Verwendung der Albedo-Domäne. Da in das Gütemaß ϵ_{MSSA} nur die mittleren Spektren eingehen, fallen die Unterschiede der Varianten, welche die Spektrenvariabilität realistischer modellieren sollen, nicht besonders hoch aus.

Um diese Unterschiede hervorzuheben, wird das Gütemaß ϵ_{AMSA} genutzt, das für den Datensatz $\mathcal{X}_{\mathrm{Q3}}$ unten in Abbildung 8.8 dargestellt wird. Dabei liefert die Datenerzeugung mit dem GCNN-Cov die besten Ergebnisse, gefolgt vom RGAN und dem GPNN. Das ist nicht verwunderlich, da diese Verfahren besonders auf eine möglichst realistische Modellierung der Spektrenvariabilität ausgelegt sind (siehe Kapitel 7). Beim RGAN bringt der Einsatz des Diskriminators bei der Datenerzeugung keinen Vorteil (RGAN-DE). Der Ansatz GPNN-M ist bei weitem am schlechtesten, weil hier nur mittlere Spektren erzeugt werden. Dazwischen reihen sich die beiden übrigen Ansätze ein, wobei hier die Verdopplung der Varianz bei der Datenerzeugung (GCNN-2) leichte Vorteile bringt. Bereits an dieser Stelle sei auf das Beispiel in Abbildung B.10 im Anhang verwiesen. Dort deckt sich der visuelle Eindruck weitestgehend mit den Ergebnissen der Gütemaße. Auch wenn es sich dabei um ein Beispiel handelt, sind die resultierenden Spektren charakteristisch für die einzelnen Verfahren.

In Abbildung 8.9 wird oben das Gütemaß ϵ_{MSSA} für den Datensatz $\mathcal{X}_{\mathrm{Q4}}$ dargestellt. Es fällt zunächst auf, dass die Werte weniger stark streuen als beim Datensatz $\mathcal{X}_{\mathrm{Q3}}$. Das trifft auch auf das Gütemaß ϵ_{AMSA} zu, das unten in Abbildung 8.9 zu sehen ist. Hier funktioniert das RGAN am besten, wobei der Einsatz des Diskriminators bei der Datenerzeugung (RGAN-DE) keinen Vorteil bringt. Wie auch beim Datensatz $\mathcal{X}_{\mathrm{Q3}}$ sind die beiden GPNN-Verfahren am schlechtesten, was darauf hindeutet, dass das kleinere Netz und der Weg über die Gauß-Prozesse insgesamt schlechter funktioniert. Das Verfahren, was beim Datensatz $\mathcal{X}_{\mathrm{Q3}}$ am besten funktioniert (GCNN-Cov), bringt hier beim Median keinen Vorteil im Vergleich zu den übrigen GCNN. Die Streuung wird sogar größer, sodass der positive Effekt spezifisch für den vorherigen Datensatz zu sein scheint. Wie auch beim vorherigen Datensatz ist der Unterschied zwischen Nutzung der Reflektanzspektren und der Spektren in der Albedo-Domäne klein.

Beim Gütemaß ϵ_{AMSA} unten in Abbildung 8.9 ergibt sich ein ähnliches Bild wie beim ersten Datensatz, der aus gefärbten Quarzsanden be-

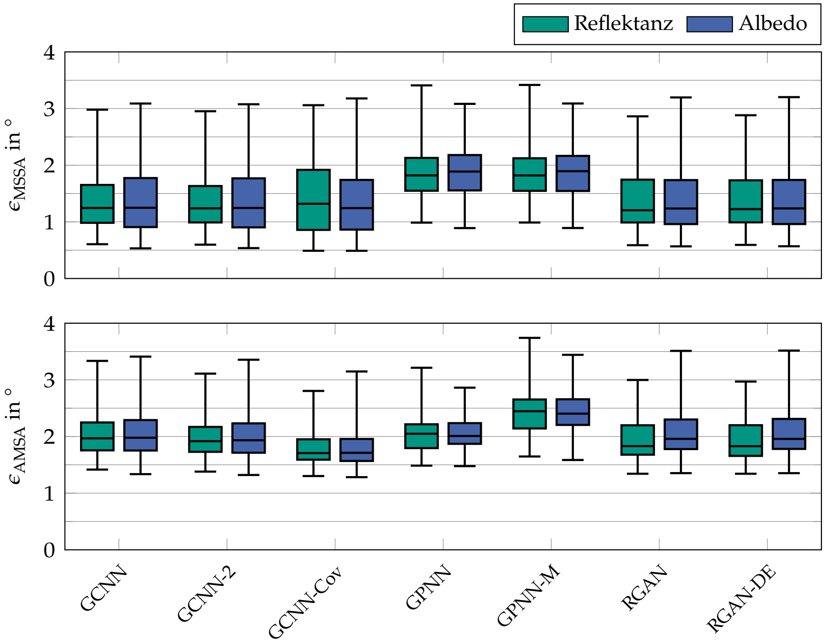


Abbildung 8.9 Darstellung von ϵ_{MSSA} und ϵ_{AMSA} als Box-Plots für den Datensatz $\mathcal{X}_{\mathrm{Q4}}$.

steht. Auch hier sind die besten Ergebnisse mit dem GCNN-Cov-Ansatz möglich, gefolgt von beiden RGAN, die im Vergleich untereinander fast identische Ergebnisse liefern. Dann kommen hier jedoch zunächst die übrigen GCNN-basierten Ansätze, wobei auch hier GCNN-2 einen kleinen Vorsprung hat. Auch hier sind die Verfahren und Varianten am besten, die eingeführt werden, um eine möglichst reale Spektrenvariabilität zu modellieren. Ausnahme ist hier das GPNN, welches schlechter als alle anderen Verfahren funktioniert. Die Variante GPNN-M ist noch schlechter, jedoch werden hier nur Mittelwertspektren erzeugt. Auch hier deckt sich der visuelle Eindruck in Abbildung B.11 im Anhang weitestgehend mit den Ergebnissen der Gütemaße.

Beim Datensatz $\mathcal{X}_{\mathrm{F4}}$ fallen beide Gütemaße, die in Abbildung 8.10 dargestellt sind, für alle Methoden schlechter aus als bei den beiden Quarzsanddatensätzen. Das gilt sowohl für den Median als auch für die

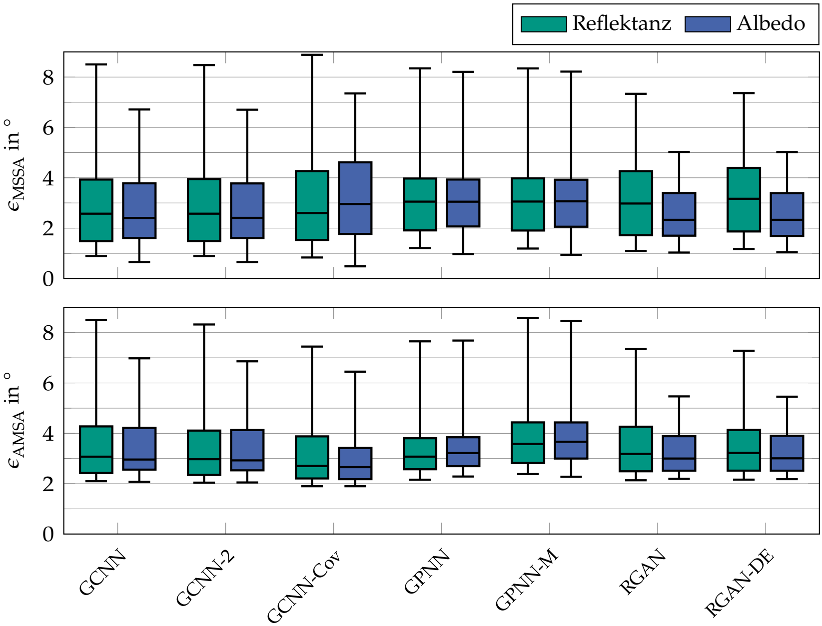


Abbildung 8.10 Darstellung von ϵ_{MSSA} und ϵ_{AMSA} als Box-Plots für den Datensatz $\mathcal{X}_{F4}$.

Streuung. Bezüglich des Gütemaßes ϵ_{MSSA} liefert das RGAN hier die besten Ergebnisse, wobei die Verwendung des Diskriminators zur Datenerzeugung auch hier zu fast identischen Ergebnissen führt. Ersteres gilt jedoch nur für die Anwendung in der Albedo-Domäne. Hier sind sowohl Median als auch Streuung vergleichsweise gering. Darauf folgen die Ansätze, die auf dem GCNN basieren, wobei hier, wie auch beim Datensatz $\mathcal{X}_{Q4}$, die Variante GCNN-Cov etwas schlechter funktioniert. Dies lässt darauf schließen, dass die Ursache dafür im größeren Abstand der Anteilsvektoren in den Trainingsdaten liegt. Auch hier funktionieren die GPNN-Verfahren am schlechtesten, jedoch mit einem geringeren relativen Unterschied als bei den beiden vorherigen Datensätzen. Mit Ausnahme der GPNN-Verfahren gibt es hier eine deutliche Verbesserung durch die Anwendung in der Albedo-Domäne, vor allem was die Streuung anbelangt. Obwohl die verwendeten KNN in der Lage sind,

nichtlineare Zusammenhänge abzubilden, ist es durchaus hilfreich, sie auf den einfacheren Mischzusammenhang (2.18) anzuwenden. Dabei ist anzumerken, dass die KNN immer noch nichtlineare Zusammenhänge lernen, da das Modell (2.18) nur unter idealen Umständen gilt, die bei den verwendeten realen Daten nicht gegeben sind.

In Abbildung 8.10 ist unten das Gütemaß ϵ_{AMSA} für den Datensatz $\mathcal{X}_{\text{F4}}$ zu sehen. Insgesamt funktioniert hier das GCNN am besten, wobei die Variante GCNN-Cov die beste ist. Dann kommt das RGAN und zuletzt das GCNN. Am schlechtesten funktioniert, wie zu erwarten, die Variante GPNN-M, weil nur Mittelwertspektren erzeugt werden. Wie auch beim Gütemaß ϵ_{MSSA} bringt die Anwendung in der Albedo-Domäne bei diesem Datensatz, mit Ausnahme des GPNNs, einen kleinen Vorteil beim Median und einen größeren bei der Streuung. Letztere ist beim RGAN und RGAN-DE am besten.

Über alle Datensätze und beide Gütemaße hinweg liefert das RGAN gute Ergebnisse, vor allem, was die Streuung anbelangt. Damit ist die Nutzung des GANs als Regularisierung durchaus sinnvoll. Dabei macht es keinen merklichen Unterschied, ob der Diskriminator zur Datenerzeugung eingesetzt wird oder nicht. Letzteres liegt auch an den verwendeten Gütemaßen. Beim Gütemaß ϵ_{MSSA} (8.2) fallen die durch den Diskriminator aussortierten Spektren bei der Bildung des mittleren Spektrums nicht ins Gewicht. Durch die Nutzung des Minimum-Operators in (8.3) werden diese beim Gütemaß ϵ_{MSSA} in den meisten Fällen nicht berücksichtigt, da andere Spektren näher an den Spektren der Testdatensätze liegen. Darüber hinaus findet auch hier eine Mittelwertbildung statt.

Besonders beim Gütemaß ϵ_{AMSA} ist die Variante GCNN-Cov am besten. Dabei ist zu beachten, dass hierfür die Kovarianzmatrizen, die mit den Anteilsvektoren korrespondieren, berechnet werden müssen. Dies ist nur dann möglich, wenn viele Spektren pro Anteilsvektor im Datensatz enthalten sind. Ein manuell erstellter Trainingsdatensatz, wie er im industriellen Umfeld notwendig ist, erfüllt diese Anforderung.

Bei allen Datensätzen sind beide Gütemaße beim GPNN am schlechtesten. Auch diese Methoden funktionieren nur bei Datensätzen, bei denen viele Spektren pro Anteilsvektor enthalten sind. Der Vorteil der Methoden ist der geringe Trainingsaufwand der KNN. Dieser ist beim GCNN und vor allem beim RGAN deutlich höher.

Die Anwendung der Verfahren in der Albedo-Domäne führt insgesamt zu recht ähnlichen Ergebnissen. Ausnahme ist der Datensatz $\mathcal{X}_{F4}$, wo eine Verbesserung erkennbar ist. Die verwendeten KNN sind also in der Lage, diesen Zusammenhang aus den Daten zu lernen, jedoch gibt es in Abhängigkeit von Datensatz und verwendetem Verfahren geringfügige Unterschiede.

In Abbildung 8.11 ist ein Beispiel für erzeugte Spektren für den Datensatz $\mathcal{X}_{F4}$ und einen Anteilsvektor zu sehen. Beispiele für die Datensätze $\mathcal{X}_{Q3}$ und $\mathcal{X}_{Q4}$ finden sich im Anhang in den Abbildungen B.10 und B.11. Dabei werden neben den 400 Originalspektren aus dem Testdatensatz für jede Methode und jede Variante 400 Spektren erzeugt. Ausnahme ist die Variante GPNN-M, wo ein Spektrum pro Anteilsvektor genügt, da hier nur mittlere Spektren genutzt werden. Auch wenn es sich dabei um ein Beispiel handelt, ist die gezeigte Spektrenvariabilität, unabhängig vom Anteilsvektor, charakteristisch für die Methoden. Die Lage der mittleren Spektren unterscheidet sich jedoch in Abhängigkeit des Anteilsvektors, sodass hier vor allem die Spektrenvariabilität und damit ϵ_{AMSA} von Interesse ist.

Zunächst ist die Spektrenvariabilität bei allen Methoden geringer, außer beim GPNN und beim GCNN-Cov. Bei beiden Methoden wird die Kovarianzmatrix der Spektren, die mit einem Anteilsvektor korrespondieren, geschätzt. Damit ist nicht verwunderlich, dass beim GCNN-Cov das Gütemaß ϵ_{AMSA} besonders gut ausfällt, auch wenn die erzeugten Spektren eine Glättung aufweisen. Diese entsteht aufgrund der zusammengesetzten Kostenfunktion, deren erster Teil zu einer Mittlung über die Spektrenmengen führt, was bei den anderen beiden Varianten noch deutlicher wird. Dort wird nur der erste Teil der Kostenfunktion genutzt. Beim GPNN täuscht der visuelle Eindruck. Das Gütemaß ϵ_{AMSA} fällt hier schlechter aus, weil auch die mittleren Spektren nicht so gut synthetisiert werden (siehe ϵ_{MSSA}). Außerdem ist im Beispiel eine zu hohe Spektrenvariabilität im unteren Wellenlängenbereich zu sehen.

Bei Methoden mit geringerer Spektrenvariabilität als in den realen Spektren aus dem Testdatensatz verhält es sich, wie in Abschnitt 7.1.3 beschrieben. Es ist beim GCNN sehr wahrscheinlich, Spektren nahe des mittleren Spektrums zu erzeugen. Durch die Verdopplung der Varianz der zufälligen Eingangsgrößen (GCNN-2) erhöht sich die Spektrenvaria-

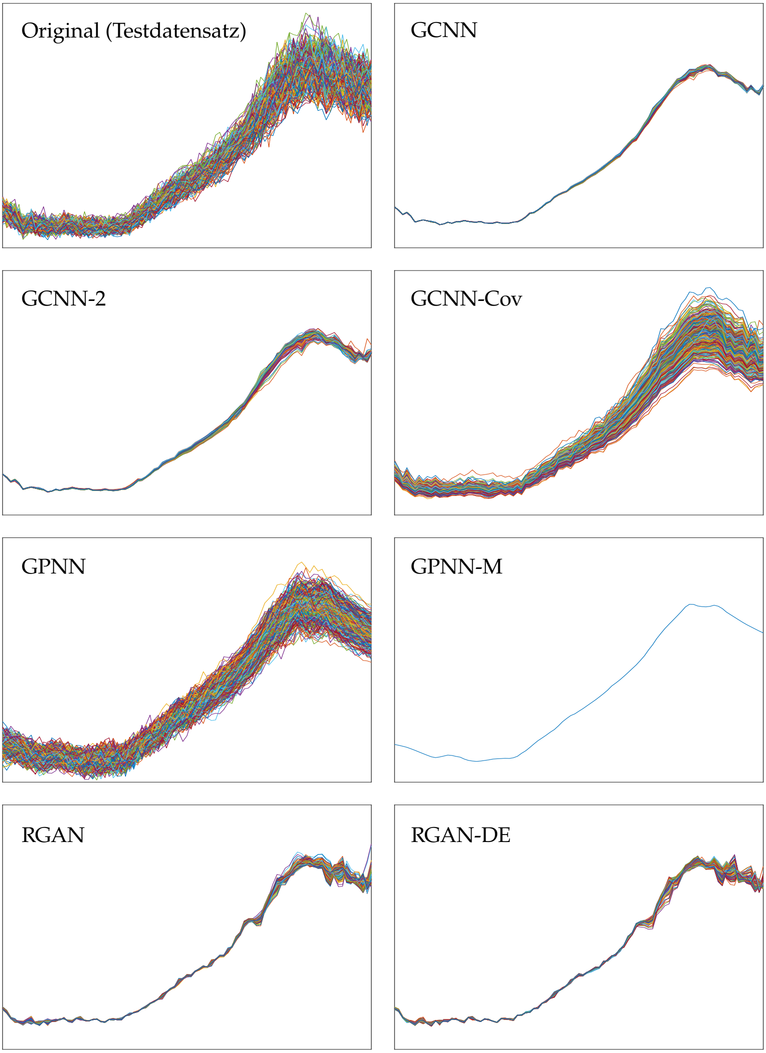


Abbildung 8.11 Beispiele für mit unterschiedlichen Verfahren und dem Anteilsvektor $\mathbf{a} = [0\,;0{,}6\,;0{,}2\,;0{,}2\,]^T$ generierte Spektren des Datensatzes $\mathcal{X}_{F4}$. Zu Gunsten der Lesbarkeit wird auf eine Achsenbeschriftung verzichtet. Die Achsen sind in allen Bildern gleich skaliert und die relative Reflektanz ist über der Wellenlänge von 450 nm bis 810 nm aufgetragen.

bilität leicht im Vergleich zur ursprünglichen Methode (GCNN). Auch die Regularisierung mit einem GAN führt zu einer ausgeprägteren Spektrenvariabilität (RGAN), auch wenn diese noch deutlich geringer ausfällt als in den Testdaten. Die Verwendung des Diskriminators während der Datenerzeugung (RGAN-DE) filtert einzelne Ausreißer heraus, was im Beispiel bei großen Wellenlängen am besten zu sehen ist.

Im nächsten Abschnitt wird unter anderem der Einfluss der Augmentierung mit den hier ausgewerteten Verfahren auf die Performanz der spektralen Entmischung untersucht. Dabei stellt sich die Frage, ob eine möglichst realistische Spektrenvariabilität von Vorteil ist oder bspw. die des GCNNs ausreicht. Denn, auch wenn es wünschenswert wäre, wird kein KNN für die spektrale Entmischung in der Lage sein, vollständig invariant gegenüber der Spektrenvariabilität zu sein. So könnte eine zu hohe Spektrenvariabilität und die damit verbundene Überlappung von Spektren, die mit ähnlichen Anteilsvektoren korrespondieren, problematisch sein. In diesem Fall wären Methoden von Vorteil, die Spektren mit einer geringeren Spektrenvariabilität erzeugen.

8.3 Spektrale Entmischung

In diesem Abschnitt werden alle in den Kapiteln 6 und 7 vorgestellten Verfahren bezüglich ihrer Performanz bei der spektralen Entmischung bewertet. Dazu wird das CNN aus Kapitel 5 mit den entsprechend erzeugten bzw. erweiterten Trainingsdaten trainiert. Zum Vergleich wird das CNN auch mit den jeweiligen ursprünglichen Trainingsdaten der Datensätze trainiert. Darüber hinaus werden die vorgestellten Verfahren auch mit Standardverfahren aus Abschnitt 2.6 verglichen.

8.3.1 Gütemaß und Darstellung der Ergebnisse

Um die Performanz der spektralen Entmischung zu bewerten, wird das quadratische Mittel des Fehlers (engl. *root-mean-square error*, RMSE) zwischen einem geschätzten Anteilsvektor $\hat{\mathbf{a}}$ und dem dazugehörigen wahren Anteilsvektor $\mathbf{a}^*$ verwendet:

$$\epsilon_{\text{RMSE}} = \sqrt{\frac{1}{P}\,\|\hat{\mathbf{a}} - \mathbf{a}^*\|_2^2}\,. \tag{8.4}$$

Dieser wird für alle Spektren bzw. Anteilsvektoren im Testdatensatz berechnet. Das Gesamtergebnis bezüglich eines Datensatzes wird als Box-Plot dargestellt. Da hier eine Eins-zu-eins-Zuordnung der Datenpunkte möglich ist, dienen die individuellen Datenpunkte als Grundlage für die Box-Plots. Um die Darstellung durch Ausreißer nicht zu sehr zu verzerren, schließen die „Antennen“ die Datenpunkte vom 3. bis zum 97. Perzentil ein. Die Box geht, wie üblich, vom 25. bis zum 75. Perzentil und der Strich gibt den Median an.

Wird der Trainingsprozess mehrfach durchgeführt, so wird für jeden Datenpunkt der Mittelwert über die Trainingsdurchläufe berechnet. Diese Mittelwertbildung wird in den Formeln nicht besonders gekennzeichnet. Bei den Verfahren aus Kapitel 6 werden jeweils 5 Wiederholungen des Trainings durchgeführt. Bei denen aus Kapitel 7 sind es 25, d. h. 5 Trainingsdurchgänge für jede Wiederholung bei der Datenerzeugung (siehe Abschnitt 8.2.3).

8.3.2 Vergleichsverfahren

In diesem Abschnitt werden die Ergebnisse der Verfahren vorgestellt, die zum Vergleich herangezogen werden. Das sind zunächst die Standardverfahren aus Kapitel 2. Diese werden stellvertretend für modellbasierte Verfahren als Vergleich herangezogen und decken auch die Verfahren aus Abschnitt 4.1 ab. Dort werden die Mischmodelle in Kombination mit KNN genutzt, um zusätzlich die Reinspektren zu ermitteln. Letztere sind hier jedoch bekannt, sodass sie bei der spektralen Entmischung verwendet werden können. Damit können direkt die modellbasierten Verfahren aus Kapitel 2 genutzt werden. Darüber hinaus wird das CNN für die spektrale Entmischung aus Kapitel 5, das nur mit den ursprünglichen Trainingsdaten trainiert wird, für den Vergleich genutzt. Die Verfahren aus Kapitel 2 benötigen keine Trainingsdaten. Bei diesen werden die Testdatensätze direkt entmischt. Bei den Verfahren, die auf dem LMM, ELMM oder GLMM basieren, und beim CNN zur spektralen Entmischung werden die Daten zusätzlich in der Albedo-Domäne entmischt. Bei den Verfahren, die auf dem FM, GBM und LQM basieren, ist dies nicht durch die Theorie motivierbar.

8.3.2.1 Konfiguration der Verfahren

Das FCLS (2.17) sowie diejenigen Verfahren, die auf dem ELMM (2.13) und GLMM (2.14) basieren, werden, wie in Abschnitt 2.6.1 beschrieben, angewendet. Die spektrale Entmischung, die auf den nichtlinearen Mischmodellen FM, GBM (2.8) und LQM (2.9) basiert, wird mit einem gradientenbasierten Liniensuchverfahren durchgeführt (vgl. Abschnitt 3.2.3.2). Dabei werden als Initialisierung, wie von Halimi et al. [41] vorgeschlagen, die Anteilsvektoren verwendet, die das FCLS ermittelt hat. Für die Verfahren, die ohne Trainingsdaten auskommen, wird jeweils eine Reinspektrenmatrix $\mathbf{M}$ benötigt. Dafür werden für jeden der P Reinstoffe die mittleren Spektren $\hat{\boldsymbol{\mu}}_p$ verwendet, die aus den Reinspektrenmengen $\mathcal{M}$ mit Hilfe von (6.5) ermittelt werden.

Das CNN für die spektrale Entmischung wird in den Ergebnissen mit CNN abgekürzt, wenn es mit den ursprünglichen, d. h. nicht augmentierten, Trainingsdaten trainiert wird. Es wird so verwendet, wie es in Abbildung 5.3 dargestellt ist. Für die Länge der eindimensionalen Faltungskerne hat sich der Wert $I = 3$ als ausreichend herausgestellt. Für die eindimensionalen *Pooling*-Schichten ist eine Größe der betrachteten Nachbarschaft von 2 sinnvoll. Für größere Werte verschwimmen die Positionen der gefundenen Merkmale zu stark, was hier nicht gewünscht ist. Die Faltungsschichten haben vom Eingang beginnend 16, 32 und 64 Merkmalskarten. Die vollständig verbundenen Schichten bestehen aus 64 und P Neuronen, wobei letzteres der Anzahl der beteiligten Reinstoffe entspricht. Als Optimierer kommt der Adam-Optimierer zum Einsatz, wobei die Parameter $\beta_1 = 0{,}9$, $\beta_2 = 0{,}999$ und $\alpha = 0{,}01$ verwendet werden. Außerdem werden auch hier die besten Ergebnisse erzielt, wenn in jedem Trainingsschritt der vollständige Trainingsdatensatz genutzt wird.

Die Anzahl an Epochen, die für das Training des CNNs nötig ist, hängt vom verwendeten Datensatz ab und davon, ob Reflektanzspektren verwendet werden oder die Spektren zuvor in die Albedo-Domäne umgerechnet werden. Sie wird in Tabelle 8.1 dargestellt. Die Unterschiede zwischen den Datensätzen kommen zustande, weil beim Datensatz $\mathcal{X}_{\mathrm{Q3}}$ die Anteilsvektoren einen kleineren Abstand zueinander haben. Dadurch setzt die Überanpassung wesentlich später ein.

Tabelle 8.1 Epochenzahl für das Training des CNNs für die spektrale Entmischung.

Domäne	Datensatz $\mathcal{X}_{Q3}$	$\mathcal{X}_{Q4}$	$\mathcal{X}_{F4}$
Reflektanz	81	21	21
Albedo	61	21	21

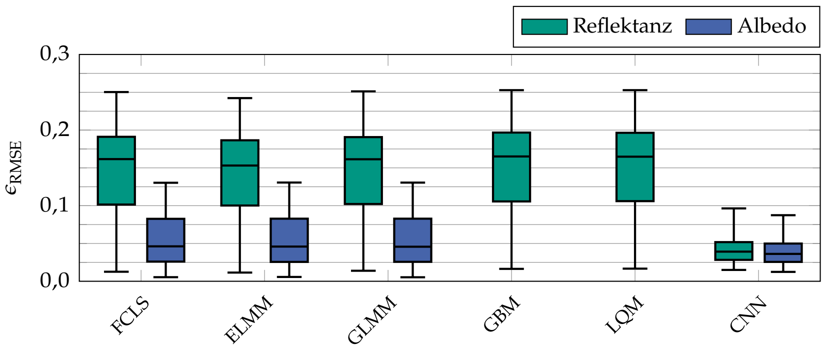


Abbildung 8.12 Darstellung von ϵ_{RMSE} als Box-Plots für den Datensatz $\mathcal{X}_{Q3}$.

8.3.2.2 Ergebnisse

Das Gütemaß ϵ_{RMSE} wird in den Abbildungen 8.12 bis 8.14 dargestellt, wobei in einer Abbildung jeweils ein Datensatz betrachtet wird. Allen Datensätzen gemein ist, dass bei den modellbasierten Verfahren die in die Albedo-Domäne transformierten Spektren wesentlich bessere Ergebnisse liefern als die Reflektanzspektren. Das liegt daran, dass es sich bei den Datensätzen um Partikelmischungen handelt und damit eine Mischung nach Abbildung 2.3 stattfindet. Beim CNN gibt es zwischen den Domänen fast keinen Unterschied. So ist es plausibel, dass das CNN die Umrechnung in die Albedo-Domäne implizit durchführt.

Bei allen Datensätzen bringen die nichtlinearen Mischmodelle GBM und LQM keinen Vorteil. Das liegt daran, dass die Optimierung der Nichtlinearitätskoeffizienten schwierig ist und kein gutes Minimum gefunden wird. Beim Datensatz $\mathcal{X}_{Q3}$ funktioniert das auf dem ELMM basierende

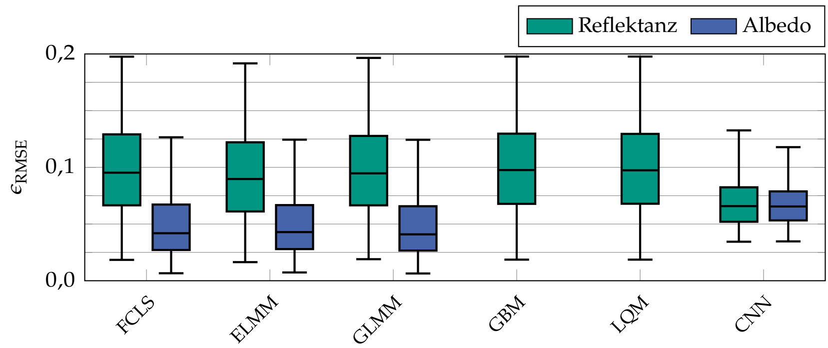


Abbildung 8.13 Darstellung von ϵ_{RMSE} als Box-Plots für den Datensatz $\mathcal{X}_{\text{Q4}}$.

Verfahren am besten von allen untersuchten modellbasierten Verfahren. Das liegt daran, dass hier auch die Spektrenvariabilität berücksichtigt wird. Durch das GLMM, was weitere Freiheitsgrade hat, kann keine Verbesserung mehr erzielt werden. Bei den Reflektanzspektren wird dadurch sogar das Ergebnis wieder verschlechtert. Am besten, vor allem bei der Streuung, aber auch beim Median, ist das CNN, welches bei diesem Datensatz genügend Trainingsdaten zur Verfügung hat, um allen betrachteten Modellen überlegen zu sein.

Beim Datensatz $\mathcal{X}_{\text{Q4}}$ fallen die Ergebnisse bei den modellbasierten Verfahren in der Albedo-Domäne ähnlich aus wie beim Datensatz $\mathcal{X}_{\text{Q3}}$, nur dass das auf dem GLMM basierende Verfahren minimal besser funktioniert. Das war zu erwarten, da beide Datensätze aus farbigen Quarzsanden desselben Herstellers bestehen. Der Datensatz $\mathcal{X}_{\text{Q4}}$ hat eine zusätzliche Farbe und die anderen drei sind leicht variiert. Bei den modellbasierten Verfahren mit Reflektanzspektren funktioniert auch hier das auf dem ELMM basierende am besten. Insgesamt fallen diese besser aus als beim Datensatz $\mathcal{X}_{\text{Q3}}$. Daraus kann geschlossen werden, dass hier die Modelle durch den zusätzlichen Reinstoff näher an der Wirklichkeit sind als beim vorherigen Datensatz. Das CNN kommt dieses Mal nicht an die besten modellbasierten Verfahren heran, auch wenn das Gütemaß ϵ_{RMSE} eine geringere Streuung aufweist. Das liegt daran, dass beim Datensatz $\mathcal{X}_{\text{Q4}}$ die Anteilsvektoren im Trainingsdatensatz weiter auseinander liegen. Dennoch ist das Resultat zufriedenstellend in Anbetracht

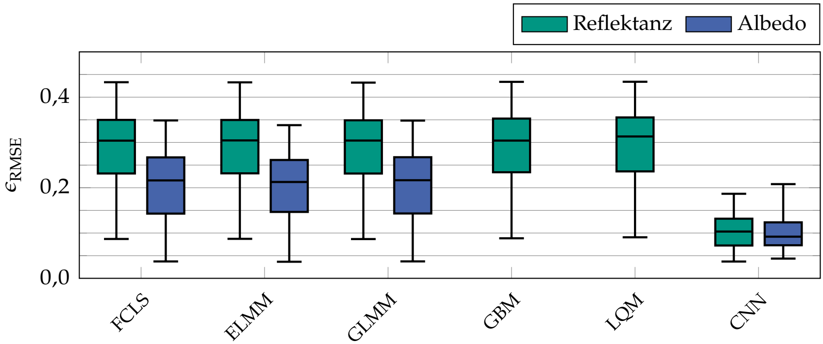


Abbildung 8.14 Darstellung von ϵ_{RMSE} als Box-Plots für den Datensatz $\mathcal{X}_{\text{F4}}$.

der Tatsache, dass kein Modellwissen benötigt wird und nur sehr wenig unterschiedliche Anteilsvektoren beim Training zur Verfügung stehen.

Beim Datensatz $\mathcal{X}_{\text{F4}}$ fällt zunächst auf, dass die Ergebnisse im Vergleich zu den Quarzsanddatensätzen insgesamt schlechter sind. Das liegt daran, dass hier mehr unbekannte Faktoren eine Rolle spielen, die zu einer höheren Spektrenvariabilität führen (siehe Abschnitt 2.5). Darüber hinaus werden die Modelle noch ungenauer, da eine höhere Varianz der Partikelgröße und -form vorliegt (siehe Abschnitt 8.1.3). Bei den modellbasierten Verfahren sehen die relativen Unterschiede zwischen den Verfahren ähnlich aus wie bei den vorhergehenden Datensätzen, nur dass der Vorteil der Verwendung der Albedo-Domäne geringer ausfällt. Dies kann mit demselben Argument begründet werden wie die schlechteren Ergebnisse im Vergleich zu den anderen Datensätzen.

Beim Datensatz $\mathcal{X}_{\text{F4}}$ ist das CNN wieder im Vorteil. Auch wenn die Resultate schlechter sind als bei den anderen Datensätzen, so sind sie, verglichen mit den anderen dargestellten Verfahren, um einiges besser und das mit denselben im Trainingsdatensatz vorhandenen Anteilsvektoren wie beim Datensatz $\mathcal{X}_{\text{Q4}}$. Das liegt daran, dass hier kein Modellwissen benötigt wird und das CNN die relevanten Zusammenhänge, die mit dem vorhandenen Wissen nicht modelliert werden können, aus den wenigen Trainingsdaten lernt. In absoluten Zahlen funktioniert die spektrale Entmischung hier trotzdem leicht schlechter als beim Datensatz $\mathcal{X}_{\text{Q4}}$,

weil für das Lernen komplexerer Zusammenhänge im Datensatz $\mathcal{X}_{F4}$ weitere Trainingsdaten notwendig wären.

Aufgrund der Ähnlichkeit der Ergebnisse in diesem Abschnitt bezogen auf die modellbasierten Verfahren werden in den folgenden Abschnitten nur das auf dem ELMM basierende Verfahren und das CNN zur spektralen Entmischung zum Vergleich herangezogen. Dabei wird sowohl das Ergebnis für Reflektanzspektren als auch für Spektren in der Albedo-Domäne dargestellt. Das ELMM wird gewählt, weil es von den modellbasierten Verfahren in den meisten Fällen am besten funktioniert. Handelt es sich in einem Fall nicht um das beste Verfahren, ist der Abstand zum besten gering. Es eignet sich für den Vergleich mit Verfahren aus Kapitel 6, bei denen die Trainingsdaten modellbasiert erzeugt werden (Ergebnisse in Abschnitt 8.3.3). Das CNN für die spektrale Entmischung, das mit den ursprünglichen Trainingsdaten trainiert wird, wie es hier in diesem Abschnitt der Fall ist, wird zum Vergleich mit den Ergebnissen der erweiterten Trainingsdatensätze genutzt (siehe Abschnitt 8.3.4). Die erweiterten Trainingsdatensätze werden ebenfalls zum Training des in Kapitel 5 vorgestellten CNNs genutzt.

8.3.3 Spektrale Entmischung mit generierten Trainingsdaten

In diesem Abschnitt werden die Trainingsdatensätze, die modellbasiert aus echten Reinspektren erzeugt werden, ausgewertet. Die Verfahren zur Datenerzeugung, die zufällige Reinspektrenwahl (ZR) und die Modellierung als normalverteilte Zufallsvektoren (NZ), werden in Kapitel 6 ausführlich dargestellt. Die Abkürzungen in den Abbildungen mit den Ergebnissen setzen sich aus der Abkürzung des Verfahrens zur Datensatzerstellung (ZR oder NZ) und der des dabei verwendeten Mischmodells (LMM, FM, GBM oder LQM) zusammen. Werden Spektren in der Albedo-Domäne erstellt und entmischt, wird die Abkürzung AD angehängt. Werden bspw. zufällige Reinspektren mit dem LMM in der Albedo-Domäne gemischt, wird dies als „ZR-LMM-AD“ bezeichnet.

Tabelle 8.2 Epochenzahl für das Training des CNNs für die spektrale Entmischung. Diese Werte ergeben sich für beide Methoden und alle Mischmodelle und werden für alle vorgegebenen Anteilsstufen genutzt.

	Datensatz		
Domäne	$\mathcal{X}_{Q3}$	$\mathcal{X}_{Q4}$	$\mathcal{X}_{F4}$
Reflektanz	251	61	31
Albedo	61	71	61

8.3.3.1 Konfiguration der Verfahren

Die Erzeugung der Trainingsdaten, die während des Trainings stattfindet, wird ausführlich in Kapitel 6 beschrieben. Bei Nutzung der Albedo-Domäne werden die Reflektanzspektren zu Beginn in die Albedo-Domäne umgerechnet.

Das CNN für die spektrale Entmischung wird so verwendet, wie es in Abschnitt 8.3.2.1 beschrieben ist. Als Trainingsdaten werden die mit den beiden Verfahren erzeugten Spektren genutzt, von denen in jedem Trainingsschritt zu jedem vorgegebenen Anteilsvektor 400 Exemplare erzeugt werden. Damit sind es genau so viele je Anteilsvektor wie im ursprünglichen Datensatz, mit dem Unterschied, dass in jeder Epoche zufällig neue Exemplare erstellt werden. Die Anteilsvektoren werden für vier unterschiedliche Anteilsstufen berechnet, die in den Ergebnissen getrennt dargestellt sind. Die konkreten Anteilsstufen hängen vom Datensatz ab und sind in den Abbildungen aufgelistet.

Die Hyperparameter bleiben dabei gleich wie in Abschnitt 8.3.2.1 mit Ausnahme der Anzahl der Epochen, wie in Tabelle 8.2 zu sehen ist. Die Anzahl der Epochen ist hier in fast allen Fällen höher als in Tabelle 8.1. Das liegt daran, dass nun mehr Datenpunkte zur Verfügung stehen und Überanpassung später auftritt. Das bedeutet jedoch nicht, dass die Resultate besser sind als beim Training mit den realen Mischspektren. Letztere bilden den wahren Mischzusammenhang genauer ab als die modellbasiert erstellten Spektren. Der Unterschied zwischen den Datensätzen mit 3 und 4 beteiligten Reinstoffen wird bereits in Abschnitt 8.3.2.1 erläutert. Hier besteht nun auch ein Unterschied zwischen den Datensätzen $\mathcal{X}_{Q4}$

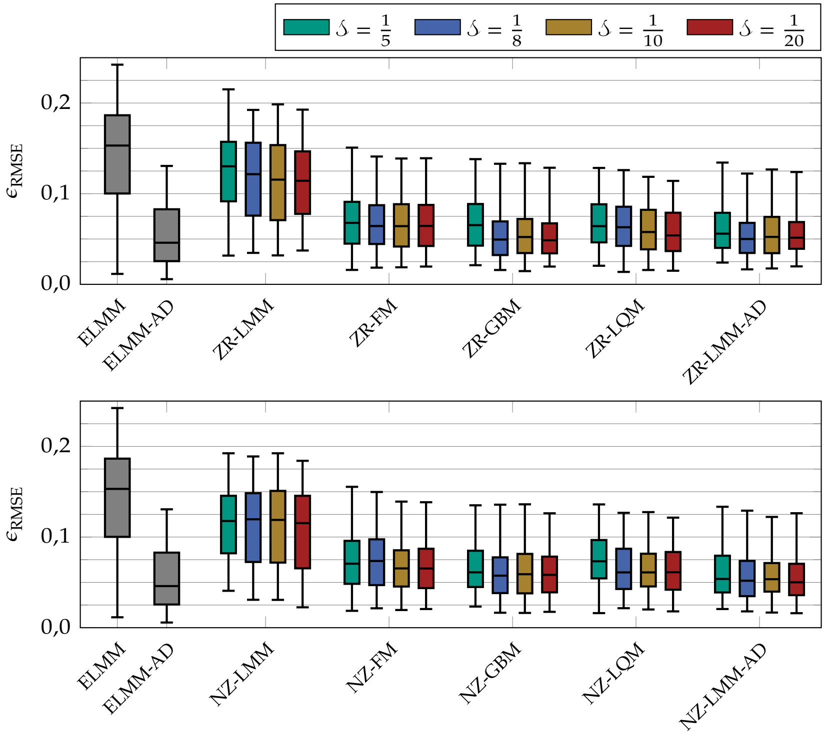


Abbildung 8.15 Darstellung von ϵ_{RMSE} als Box-Plots für den Datensatz $\mathcal{X}_{\mathrm{Q3}}$. Dabei sind oben die Ergebnisse der Datenerzeugung mit zufälliger Reinspektrenwahl und unten die der Datenerzeugung durch Modellierung als normalverteilte Zufallsvektoren zu sehen.

und $\mathcal{X}_{\mathrm{F4}}$, was daran liegt, dass bei letzterem die Modelle die Realität schlechter beschreiben (siehe Abschnitt 8.1.3).

8.3.3.2 Ergebnisse

In Abbildung 8.15 sind die Ergebnisse für den Datensatz $\mathcal{X}_{\mathrm{Q3}}$ zu sehen. Dabei fällt bei der zufälligen Reinspektrenwahl zunächst auf, dass bei der direkten spektralen Entmischung mit dem ELMM-AD sehr gute Ergebnisse erzielt werden, die durch das vorgestellte Verfahren der zufälligen Reinspektrenwahl nur minimal in Bezug auf die Streuung verringert

werden können (ZR-LMM-AD). Hier ist das verwendete Modell genau genug, dass die Spektrenvariabilität durch die Skalierungsfaktoren im ELMM (2.13) näherungsweise abgebildet werden kann.

Bei Verwendung der Reflektanzspektren ergibt sich bei allen Mischmodellen (vgl. Abbildung 8.12) eine deutliche Verbesserung der Ergebnisse, wobei das FM als Spezialfall des GBMs zu sehen ist. Dabei sticht das FM heraus, weil es ohne zusätzliche Nichtlinearitätskoeffizienten auskommt. Am besten funktioniert hier das GBM. So können die Modelle aus Abschnitt 2.4.1, die auf Mehrfachstreuung beruhen, recht gute Ergebnisse liefern. Die Nichtlinearitätskoeffizienten werden jedoch nicht aufgrund physikalischer Gegebenheiten gesetzt, sondern in einem Optimierungsprozess bestimmt. Diese Nichtlinearitätskoeffizienten sind im Anhang in Tabelle B.1 für die untersuchten Verfahren und Anteilsstufen aufgelistet. Neben den Nichtlinearitätskoeffizienten, die mit Hilfe der Validierungsdaten bestimmt und für die Ergebnisse auf die Testdaten angewandt werden, sind dort auch diejenigen gezeigt, die bei einer Optimierung mit dem Testdatensatz resultieren würden. Diese dürfen nicht zur Bewertung bezüglich der spektralen Entmischung verwendet werden. Allerdings kann der Vergleich einen Hinweis darauf geben, ob mit den Validierungsdaten geeignete Werte ermittelt worden sind. Bei der Betrachtung der Werte fällt auf, dass sie sich in Abhängigkeit der Anteilsstufe unterscheiden. Das liegt daran, dass nur ein Wert für alle Mischungen verwendet wird und das Optimum von den verwendeten Anteilsvektoren abhängt. Darüber hinaus fällt auf, dass beim LQM sich die mit den Validierungsdaten ermittelten Werte häufiger mit den für den Testdatensatz optimalen decken. Bei den Ergebnissen in Abbildung 8.15 ist der Einfluss der Anteilsstufe $\mathcal{S}$ relativ gering. In den meisten Fällen tendieren kleinere Anteilsstufen zu leicht besseren Ergebnissen.

In Abbildung 8.15 sind auch die Ergebnisse der Modellierung als normalverteilte Zufallsvariablen für den Datensatz $\mathcal{X}_{Q3}$ zu sehen. Diese sind denen der zufälligen Reinspektrenwahl sehr ähnlich. Damit kann bestätigt werden, dass die getroffene Annahme der Normalverteilung gerechtfertigt ist, wenn das Ziel die spektrale Entmischung ist. Dabei sind die Ergebnisse der linearen Modelle insgesamt geringfügig besser, die der nichtlinearen geringfügig schlechter als bei der zufälligen Reinspektrenwahl. Dies kann damit erklärt werden, dass letztere auch

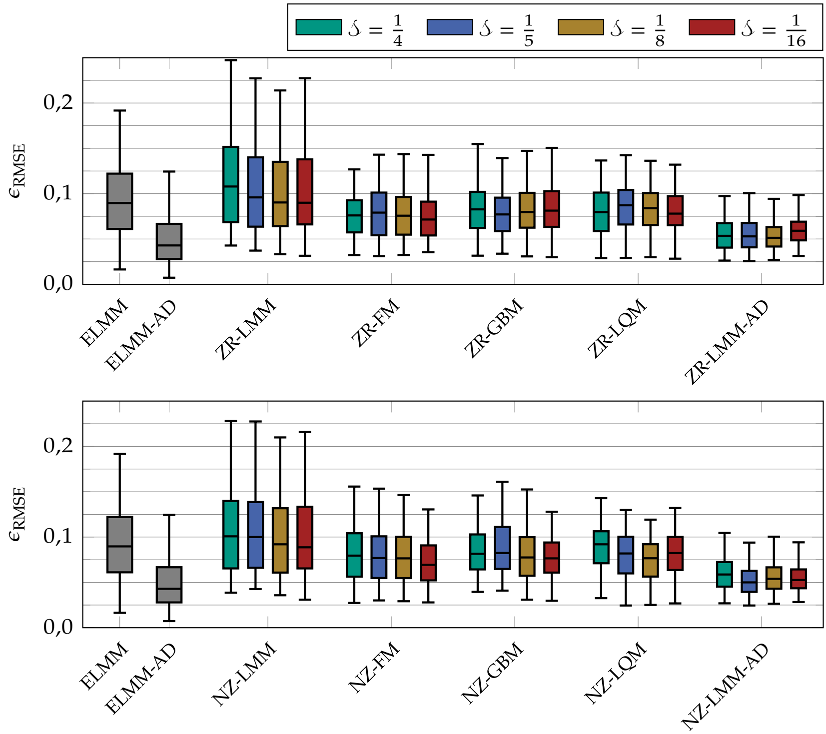


Abbildung 8.16 Darstellung von ϵ_{RMSE} als Box-Plots für den Datensatz $\mathcal{X}_{\text{Q4}}$. Dabei sind oben die Ergebnisse der Datenerzeugung mit zufälliger Reinspektrenwahl und unten die der Datenerzeugung durch Modellierung als normalverteilte Zufallsvektoren zu sehen.

Produkte von Reinspektren enthalten. Im Gegensatz zu Summen sind Produkte normalverteilter Zufallsvektoren im Allgemeinen nicht normalverteilt. Es wird jedoch angenommen, dass die Spektrenvariabilität von Mischspektren ebenfalls durch eine Normalverteilung ausreichend genau modelliert werden kann (siehe Abschnitt 6.2).

In Abbildung 8.16 sind die Ergebnisse für den Datensatz $\mathcal{X}_{\text{Q4}}$ zu sehen. Bei diesem Datensatz können bei der zufälligen Reinspektrenwahl im Großen und Ganzen die gleichen Beobachtungen gemacht werden wie beim vorherigen Datensatz. Das liegt daran, dass es sich dabei um Quarzsande des gleichen Herstellers handelt. Der größte Unterschied

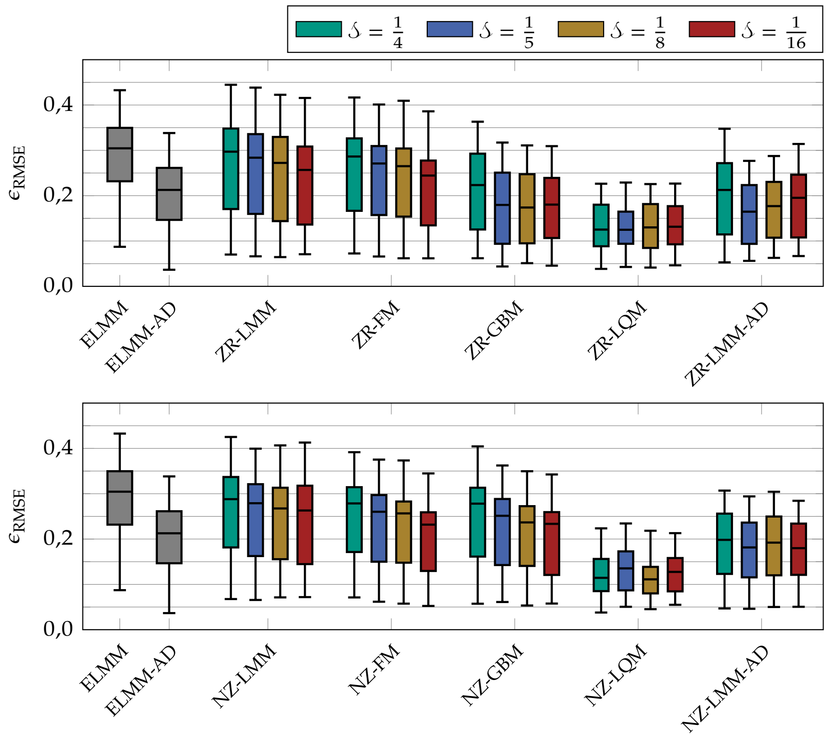


Abbildung 8.17 Darstellung von ϵ_{RMSE} als Box-Plots für den Datensatz $\mathcal{X}_{F4}$. Dabei sind oben die Ergebnisse der Datenerzeugung mit zufälliger Reinspektrenwahl und unten die der Datenerzeugung durch Modellierung als normalverteilte Zufallsvektoren zu sehen.

ist, dass das direkte Entmischen mit dem ELMM besser funktioniert (vgl. Abschnitt 8.3.2.2). Auch im Vergleich mit der Modellierung als normalverteilte Zufallsvektoren können die gleichen Beobachtungen gemacht werden. Dies bestätigt, zumindest in Bezug auf die Quarzsande, die Aussagen bezüglich der Annahme der Normalverteilung. Eine Auflistung der Nichtlinearitätskoeffizienten von GBM und LQM ist im Anhang in Tabelle B.2 zu sehen.

In Abbildung 8.17 sind die Ergebnisse für den Datensatz $\mathcal{X}_{F4}$ zu sehen. Hier können bei der zufälligen Reinspektrenwahl im Vergleich mit den korrespondierenden direkten Verfahren zur spektralen Entmischung in

Abbildung 8.14 insgesamt bessere Ergebnisse erzielt werden. Im Unterschied zu den vorherigen Datensätzen funktioniert hier ein Verfahren am besten, das die Reflektanzspektren nutzt. Der Trainingsdatensatz, der basierend auf dem LQM gemischt wurde, führt zu besseren Ergebnissen als derjenige, der in der Albedo-Domäne mit dem LMM gemischt wurde. Dafür kommen zwei Ursachen in Frage: Zum einen ist das Modell zur Umrechnung in die Albedo-Domäne aufgrund der Varianz der Partikelgröße und der raueren Oberfläche (siehe Abbildung 8.2) hier ungenauer. Zum anderen führt die rauere Oberfläche auch zu Mehrfachstreuung, wie sie in Abbildung 2.2(b) schematisch dargestellt wird. Das LQM modelliert genau diese, wobei es im Gegensatz zum GBM Mehrfachstreuung am gleichen Reinstoff modelliert. Insgesamt gilt aber zu beachten, dass die Ergebnisse beim Datensatz $\mathcal{X}_{\mathrm{F4}}$ schlechter ausfallen als bei den Quarzsanddatensätzen.

Die Wahl der Anteilsstufe hat hier unterschiedlich großen Einfluss, je nachdem, welches Modell zugrunde gelegt wird. Insgesamt führen auch hier kleinere Anteilsstufen zu besseren Ergebnissen. Dies ist wenig überraschend, da dadurch mehr Anteilsvektoren im Trainingsdatensatz zur Verfügung stehen. Jedoch führen zu viele zu einer Überlappung der Spektren unterschiedlicher Anteilsvektoren, was wiederum zu schlechteren Ergebnissen bei der Entmischung führen kann.

In Abbildung 8.17 sind auch die Ergebnisse der Modellierung als normalverteilte Zufallsvektoren für den Datensatz $\mathcal{X}_{\mathrm{F4}}$ zu sehen. Auch bei diesem Datensatz gilt, dass die Ergebnisse denen der zufälligen Reinspektrenwahl sehr ähnlich sind. Dabei sind sie hier für die meisten Modelle leicht besser. Ausnahme ist das GBM, wo die Ergebnisse deutlich schlechter werden. Das lässt sich damit erklären, dass der Nichtlinearitätskoeffizient, der mit den Validierungsdaten bestimmt wird (siehe Abschnitt 8.1.5), bei den Testdaten zu schlechteren Ergebnissen führt. Eine Auflistung der Nichtlinearitätskoeffizienten von GBM und LQM ist im Anhang in Tabelle B.3 zu sehen. Dort wird deutlich, dass beim GBM die Bestimmung der Nichtlinearitätskoeffizienten bei der zufälligen Reinspektrenwahl besser funktioniert als bei der Modellierung als normalverteile Zufallsvektoren. Beim LQM funktioniert die Bestimmung für beide Verfahren gut.

Im Anhang B.1 sind zur besseren Vergleichbarkeit die Box-Plots aller Verfahren zur modellbasierten Datenerzeugung gemeinsam für die Anteilsstufen $\mathcal{S} = \frac{1}{10}$ bzw. $\mathcal{S} = \frac{1}{8}$ dargestellt. Zusätzlich werden für diese Anteilsstufen im Anhang B.2 die Ergebnisse für einige Verfahren als Histogramme dargestellt. Damit kann ein genauerer Eindruck über die Verteilung von ϵ_{RMSE} gewonnen werden.

Die vorgestellten Verfahren zur Datenerzeugung sind also in der Lage, zusammen mit einem CNN bessere Ergebnisse zu erzielen als ihre korrespondierenden direkten Verfahren zur spektralen Entmischung. Dabei hängt die Verbesserung von den Daten und vom genutzten Modell ab. Bei den Quarzsanddatensätzen ist, verglichen mit der Entmischung mit dem ELMM in der Albedo-Domäne, keine Verbesserung mehr möglich, weil das Modell hier schon sehr genau ist.

Des Weiteren liefern beide Verfahren ähnlich gute Ergebnisse, woraus geschlossen werden kann, dass bei der Modellierung die Annahme normalverteilter Zufallsvektoren getroffen werden kann. Zumindest ist die Modellierung genau genug, um Trainingsdaten für die spektrale Entmischung zu erzeugen.

8.3.4 Spektrale Entmischung mit erweiterten Trainingsdaten

In diesem Abschnitt werden die Auswirkungen der Verfahren zur Datenaugmentierung, die in Kapitel 7 ausführlich beschrieben werden, vorgestellt und ausgewertet. Die hier verwendeten Abkürzungen für die Methoden, die in den Abbildungen zu finden sind, werden in Abschnitt 8.2 eingeführt.

8.3.4.1 Konfiguration der Verfahren

Die Erzeugung der zusätzlichen Trainingsdaten erfolgt analog zu Abschnitt 8.2.2. Das Training der KNN, die die Datenerzeugung durchführen, erfolgt mit den gleichen Parametern. Bei der Erzeugung der zusätzlichen Trainingsdaten werden Anteilsvektoren vorgegeben, die auf jeweils 4 Anteilsstufen basieren. Die Anteilsvektoren werden analog zu denen in Abschnitt 8.1 erstellt, wobei hier die genutzten Anteilsstufen abhängig vom Datensatz sind und in den Abbildungen angegeben

Tabelle 8.3 Epochenzahl für das Training des CNNs für die spektrale Entmischung. Diese Werte ergeben sich für Reflektanz- und Albedo-Domäne und werden für alle vorgegebenen Anteilsstufen genutzt.

	Datensatz		
Verfahren	$\mathcal{X}_{Q3}$	$\mathcal{X}_{Q4}$	$\mathcal{X}_{F4}$
GCNN	251	51	21
GCNN-2	251	51	21
GCNN-Cov	251	51	21
GPNN	251	41	31
GPNN-M	251	31	21
RGAN	251	51	21
RGAN-DE	251	51	21

werden. Die unterschiedlichen Anteilsstufen werden aufgrund der unterschiedlichen bereits vorhandenen Anteilsstufen der Datensätze verwendet. Anteilsvektoren, die bereits im ursprünglichen Trainingsdatensatz vorkommen, werden nicht benutzt, da hier bereits Spektren vorhanden sind. Zur Augmentierung werden die ursprünglichen Trainingsdatensätze mit den erzeugten Spektren ergänzt. Auch hier werden 400 Spektren je Anteilsvektor erzeugt, damit die Anzahl der in den ursprünglichen Trainingsdaten entspricht.

Das CNN für die spektrale Entmischung wird so verwendet, wie in Abschnitt 8.3.2.1 beschrieben, wobei die Hyperparameter bis auf die Anzahl der Epochen gleich bleiben. Letztere sind in Tabelle 8.3 zu finden. Dabei fällt auf, dass die Anzahl der Epochen höher ist als in Tabelle 8.1, wo die ursprünglichen Trainingsdatensätze genutzt werden. Das liegt daran, dass die erweiterten Trainingsdatensätze mehr Datenpunkte beinhalten und später zu einer Überanpassung führen. Bis auf eine Ausnahme ist dies beim Datensatz $\mathcal{X}_{F4}$ nicht der Fall. Wie bereits erwähnt, hat dieser die meisten Unsicherheiten bei der Modellierung. Die Ziffer 1 bei den Werten kommt daher, dass in Zehnerschritten, beginnend bei 1, nach der Anzahl gesucht wird. Dies hat sich als ausreichend genau herausgestellt.

8.3.4.2 Ergebnisse

Zusätzlich zu den hier dargestellten Box-Plots werden für die Anteilsstufen $\mathcal{S} = \frac{1}{10}$ bzw. $\mathcal{S} = \frac{1}{8}$ im Anhang B.2 für einige Verfahren die Ergebnisse als Histogramme dargestellt. Damit kann ein genauerer Eindruck von der Verteilung von ϵ_{RMSE} gewonnen werden.

In Abbildung 8.18 sind die Ergebnisse der spektralen Entmischung für den Datensatz $\mathcal{X}_{\text{Q3}}$ dargestellt. Bei den Resultaten zum GCNN fällt zunächst auf, dass in allen Fällen eine Verbesserung gegenüber dem nicht erweiterten Datensatz (CNN) vorliegt, außer wenn die Schrittweite mit $\mathcal{S} = \frac{1}{5}$ zu groß gewählt wird. Dies gilt auch für das GPNN und das RGAN. Hier genügt die Anzahl der zusätzlichen Anteilsvektoren nicht, um eine Verbesserung zu erzielen. Es findet sogar eine Verschlechterung statt, was am Fehler des aus den Daten gelernten Modells des GCNNs liegt. Bei kleineren Anteilsstufen überwiegt der Vorteil durch die zusätzlichen Daten. Zwischen den anderen untersuchten Anteilsstufen besteht kaum ein Unterschied.

Die Variante, die eine Regularisierung mit Hilfe der Kovarianzmatrix durchführt (GCNN-Cov), hat in der Reflektanz-Domäne einen kleinen Vorsprung vor den anderen Varianten. Daraus kann geschlossen werden, dass die realistischere Modellierung der Spektrenvariabilität bei diesem Datensatz vorteilhaft ist.

In Abbildung 8.18 ist zu sehen, dass mit dem GPNN ähnlich gute Ergebnisse erzielt werden können wie mit dem GCNN. Bei ähnlichem Median ist die Streuung der Werte in der Box geringer. Dabei genügt die Verwendung der Mittelwertspektren (GPNN-M). Damit zeigt sich, dass auch mit den hier verwendeten einfachen KNN gute Ergebnisse in der spektralen Entmischung erzielt werden können, obwohl diese im direkten Vergleich mit den Testspektren in Abschnitt 8.2 am schlechtesten abschneiden. Insbesondere folgt daraus, dass auch ohne Modellierung der Spektrenvariabilität eine Verbesserung erzielt werden kann und dass die beim GPNN verwendete Modellierung der Kovarianzfunktion kaum Vorteile bringt.

Beim RGAN fallen die Ergebnisse, die in Abbildung 8.18 dargestellt sind, ähnlich aus. Hier ist jedoch die Streuung, die durch die „Antennen“ in den Box-Plots dargestellt wird, nach oben hin geringer. Damit ergibt sich ein Vorteil der Regularisierung mit dem Diskriminator. Die

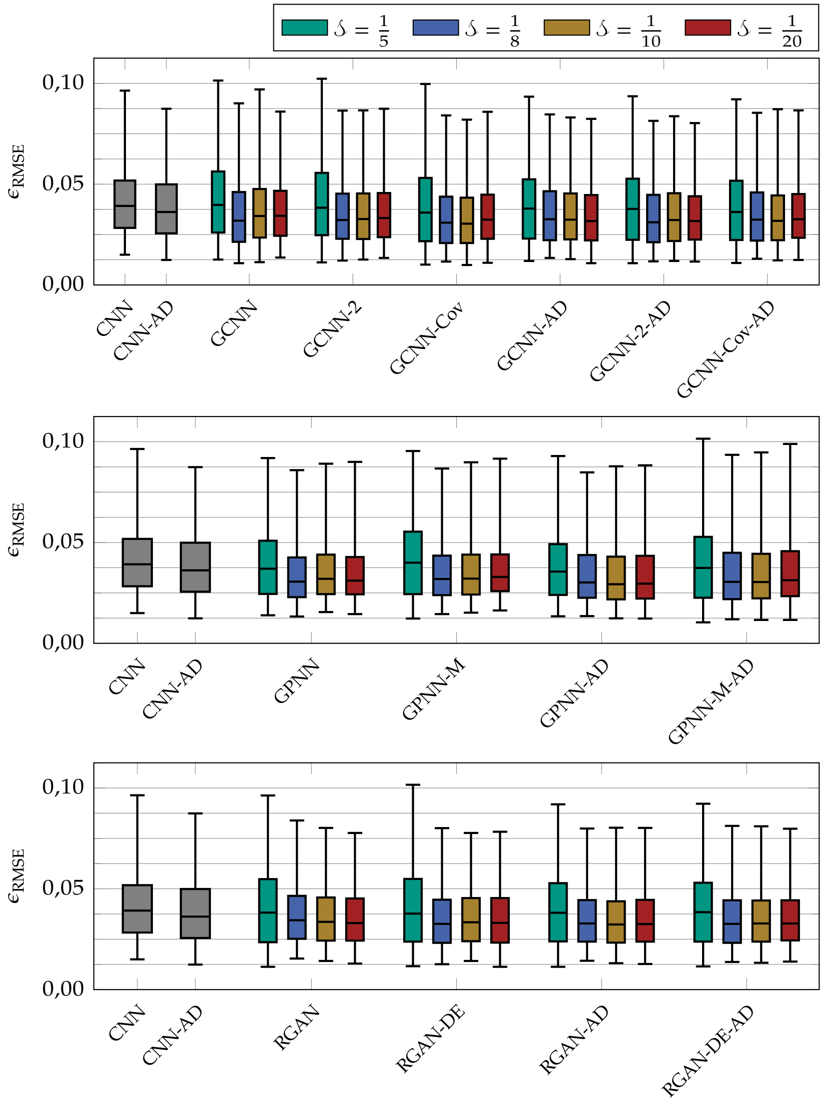


Abbildung 8.18 Darstellung von ϵ_{RMSE} als Box-Plots für den Datensatz $\mathcal{X}_{\mathrm{Q3}}$. Dabei sind die Ergebnisse der Augmentierung mit dem GCNN oben, mit dem GPNN in der Mitte und mit dem RGAN unten zu sehen.

Verwendung des Diskriminators bei der Datenerzeugung hat hier keinen nennenswerten Einfluss.

Die Unterschiede zwischen Reflektanz- und Albedo-Domäne sind bei allen Verfahren gering, sodass die Aussage bestätigt werden kann, dass die KNN diesen Zusammenhang implizit lernen. Im Anhang in Abbildung B.2 sind zur besseren Vergleichbarkeit die Box-Plots aller augmentierenden Verfahren nebeneinander für die Anteilsstufe $\mathcal{S} = \frac{1}{10}$ dargestellt.

In Abbildung 8.19 sind die Ergebnisse für den Datensatz $\mathcal{X}_{Q4}$ zu sehen. Auch hier findet bei den Resultaten zum GCNN in jedem untersuchten Fall eine Verbesserung im Vergleich zum Training mit dem nicht augmentierten Trainingsdatensatz (CNN) statt. Dabei ist zu beachten, dass auch keines dieser Verfahren in der Lage ist, die direkte spektrale Entmischung mit dem ELMM zu schlagen. Im Anhang in Abbildung B.4 sind zur besseren Vergleichbarkeit die Box-Plots aller augmentierenden Verfahren gemeinsam für die Anteilsstufe $\mathcal{S} = \frac{1}{8}$ dargestellt. Dort ist auch die direkte spektrale Entmischung mit dem ELMM zum Vergleich dargestellt. Diese Aussagen gelten auch für das GPNN und das RGAN. Das ist, wie auch an anderer Stelle, damit zu erklären, dass im Vergleich zum vorherigen Datensatz ein größerer Abstand zwischen den Anteilsvektoren im Trainingsdatensatz vorliegt. Dennoch kann auch hier die Performanz des CNNs durch Augmentierung verbessert werden.

Ebenfalls für alle Verfahren gilt auch beim Datensatz $\mathcal{X}_{Q4}$, dass der Unterschied zwischen Reflektanz- und Albedo-Domäne gering ist, was mit den bisherigen Erkenntnissen übereinstimmt. Im Gegensatz zum Datensatz $\mathcal{X}_{Q3}$ bringt hier auch die größte Anteilsstufe eine Verbesserung und der Unterschied zur nächst kleineren ist geringer. Tendenziell gilt hier, dass kleinere Anteilsstufen zu besseren Ergebnissen führen, jedoch gibt es dabei einige Unregelmäßigkeiten. Dies ist damit zu erklären, dass im ursprünglichen Trainingsdatensatz die Datenpunkte bezüglich der Anteilsvektoren einen größeren Abstand haben. Daher bringen schon wenige zusätzliche Daten, trotz Modellfehler im datenerzeugenden KNN, einen Vorteil, der mit steigender Anzahl an zusätzlichen Datenpunkten größer wird.

Das GPNN schneidet hier im Vergleich mit den anderen Augmentierungsverfahren am schlechtesten ab, aber auch hier findet eine Verbes-

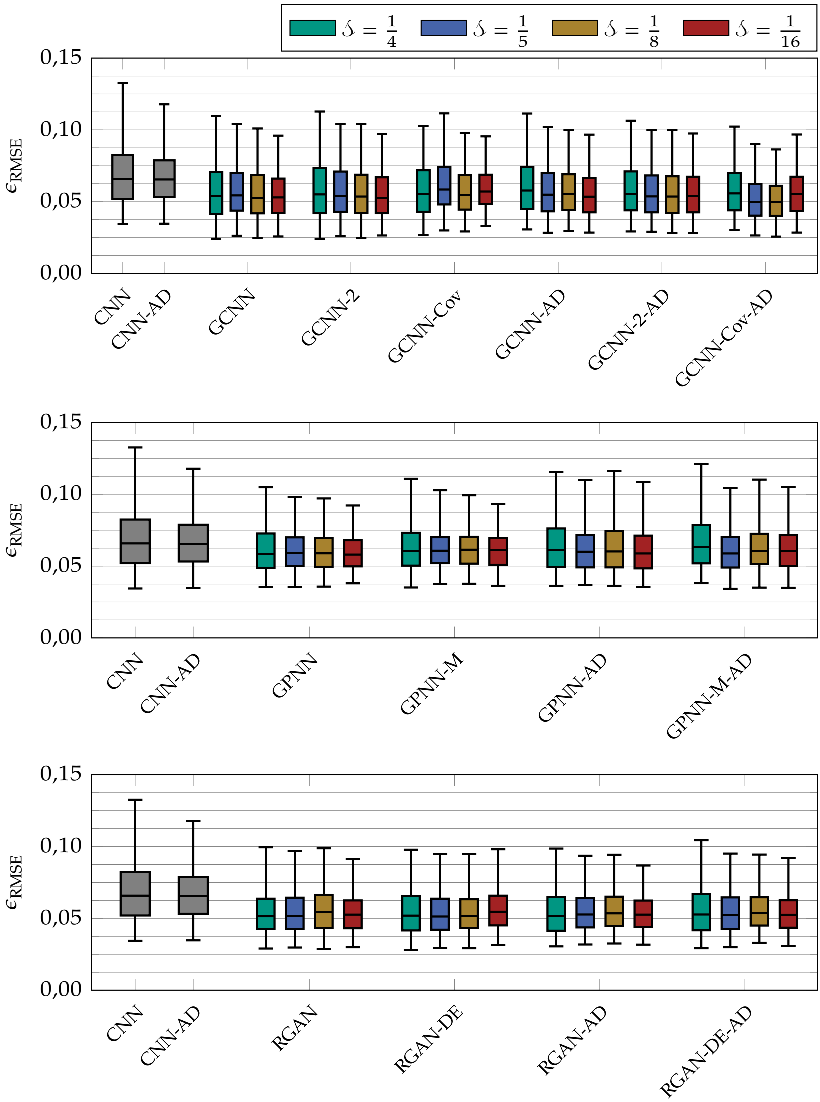


Abbildung 8.19 Darstellung von ϵ_{RMSE} als Box-Plots für den Datensatz $\mathcal{X}_{Q4}$. Dabei sind die Ergebnisse der Augmentierung mit dem GCNN oben, mit dem GPNN in der Mitte und mit dem RGAN unten zu sehen.

serung gegenüber der Referenz (CNN) statt. Wie beim Datensatz $\mathcal{X}_{Q3}$ ist hier die Variante GPNN-M ähnlich gut wie die, die auch die Kovarianzfunktion berücksichtigt (GPNN). Allerdings schneiden das GCNN und das RGAN deutlich besser ab. Bei größerem Abstand zwischen den Anteilsvektoren können die verwendeten kleinen GPNN die Zusammenhänge nicht so gut abbilden wie die anderen vorgestellten KNN.

Das RGAN schneidet bei diesem Datensatz, vor allem bei der Streuung, am besten ab. Der Unterschied ist am deutlichsten bei großen Anteilsstufen. Die Regularisierung mit dem Diskriminator zahlt sich auch beim Datensatz $\mathcal{X}_{Q4}$ aus. Damit ergibt sich hier die geringste Abhängigkeit von der Anteilsstufe bei diesem Datensatz. Die Verwendung des Diskriminators bei der Datenerzeugung (RGAN-DE) führt zu keiner zusätzlichen Verbesserung. Die dadurch aussortierten erzeugten Spektren scheinen nicht ins Gewicht zu fallen.

In Abbildung 8.20 sind die Ergebnisse der spektralen Entmischung für den Datensatz $\mathcal{X}_{F4}$ zu sehen. Wie bei den bisherigen Ergebnissen sind diese im Vergleich zu den anderen Datensätzen beim Datensatz $\mathcal{X}_{F4}$ insgesamt schlechter. Eine Verbesserung im Vergleich zum CNN, das mit dem nicht augmentierten Trainingsdatensatz trainiert wird, ist bei allen Verfahren und Anteilsstufen gegeben. Dabei funktioniert letzteres besser als alle modellbasierten Verfahren. Ein Vergleich aller Verfahren findet sich im Anhang in Abbildung B.6.

Wie bei den vorherigen Datensätzen ist der Einfluss der Anteilsstufe gering, wobei kleinere Anteilsstufen tendenziell zu besseren Ergebnissen führen. Im Gegensatz zum Datensatz $\mathcal{X}_{Q3}$ ist auch hier, wie beim Datensatz $\mathcal{X}_{Q4}$, die größte Anteilsstufe nicht schlechter als die Referenz. Damit bestätigt sich die Vermutung, dass dies mit den verfügbaren Anteilsvektoren im ursprünglichen Trainingsdatensatz zusammenhängt, die beim Datensatz $\mathcal{X}_{Q3}$ anders ist als bei den restlichen Datensätzen. Ebenfalls wie bei den anderen Datensätzen zeigt sich hier der geringe Unterschied zwischen Reflektanz- und Albedo-Domäne.

Am besten funktioniert das GCNN, wobei die Regularisierung mit der Kovarianzmatrix einen leichten Vorsprung hat. Das RGAN liefert minimal schlechtere Ergebnisse. Dies ist bei den Quarzsanddatensätzen nicht der Fall gewesen. Das zeigt, dass die Regularisierung mit dem Diskriminator nicht immer zu besseren Ergebnissen führt, wobei der

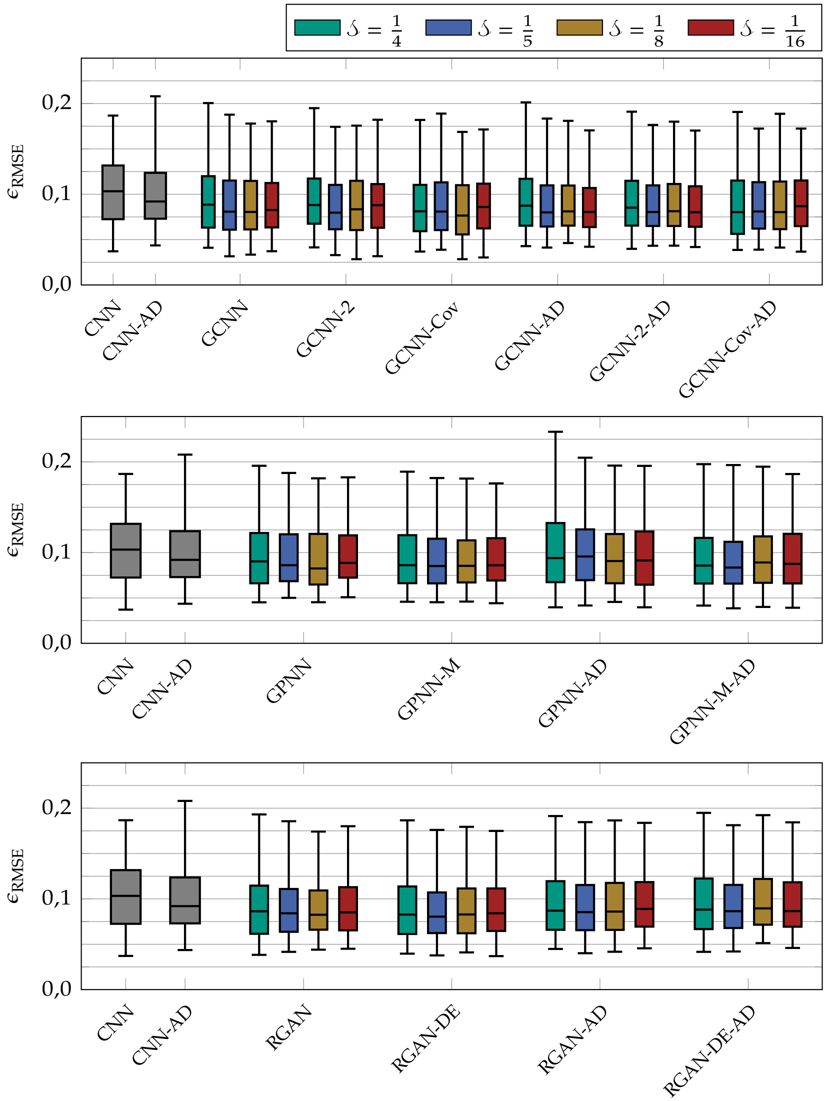


Abbildung 8.20 Darstellung von ϵ_{RMSE} als Box-Plots für den Datensatz $\mathcal{X}_{\mathrm{F4}}$. Dabei sind die Ergebnisse der Augmentierung mit dem GCNN oben, mit dem GPNN in der Mitte und mit dem RGAN unten zu sehen.

Datensatz $\mathcal{X}_{\mathrm{F4}}$ am anspruchsvollsten ist (siehe Abschnitt 8.1.3). Auch hier findet keine Verbesserung der Ergebnisse durch die Verwendung des Diskriminators bei der Datenerzeugung statt.

Am schlechtesten funktioniert, wie auch beim Datensatz $\mathcal{X}_{\mathrm{Q4}}$, das GPNN. Damit liegt die Vermutung nahe, dass dieses Problem mit dem größeren Abstand der Anteilsvektoren im ursprünglichen Trainingsdatensatz zusammenhängt. Das Weglassen der Kovarianzfunktion (GPNN-M) führt hier sogar zu leicht besseren Ergebnissen, was nahelegt, dass die Modellierung der Spektrenvariabilität zu ungenau ist, um einen Vorteil zu bringen.

8.3.5 Vergleich Reflektanz- und Albedo-Domäne

Wie bereits in den vorherigen Abschnitten festgestellt wurde, fällt sowohl bei der spektralen Entmischung (siehe Abschnitt 8.3.4.2) als auch bei der Datenerzeugung (siehe Abschnitt 8.2.3) mit KNN der Unterschied zwischen der Verwendung von Reflektanzspektren und Spektren in der Albedo-Domäne gering aus. Darüber hinaus gehen die Unterschiede in beide Richtungen. Dies variiert in Abhängigkeit des verwendeten Datensatzes, des verwendeten Verfahrens und der verwendeten Anteilsstufe, wobei keine Regelmäßigkeit festzustellen ist.

Durch die modellbasierten Verfahren ist bekannt, dass die Mischspektren der untersuchten Datensätze in der Albedo-Domäne mit dem linearen Mischmodell gut beschrieben werden können (siehe Abschnitte 8.3.2.2 und 8.3.3.2). Damit ist die Funktion, die das KNN realisiert, bei Spektren in der Albedo-Domäne einfacher als bei Reflektanzspektren. Daher stellt sich die Frage, ob dies einen Vorteil bei weniger verfügbaren Trainingsdaten darstellt.

Dies wird in Abbildung 8.21 für den Datensatz $\mathcal{X}_{\mathrm{Q3}}$ untersucht. Dieser eignet sich dafür am besten, da hier die meisten unterschiedlichen Werte in den Anteilsvektoren im ursprünglichen Trainingsdatensatz enthalten sind. Dargestellt wird das Gütemaß ϵ_{RMSE}. Es wird das CNN aus Kapitel 5 mit dem ursprünglichen Trainingsdatensatz trainiert, bei dem nach und nach Anteilsvektoren und alle dazugehörigen Spektren entfernt werden. Die Hyperparameter werden wie in Abschnitt 8.3.2.1 belassen. Die Ausnahme ist die Anzahl an Epochen, die angepasst werden muss (siehe Tabelle 8.4).

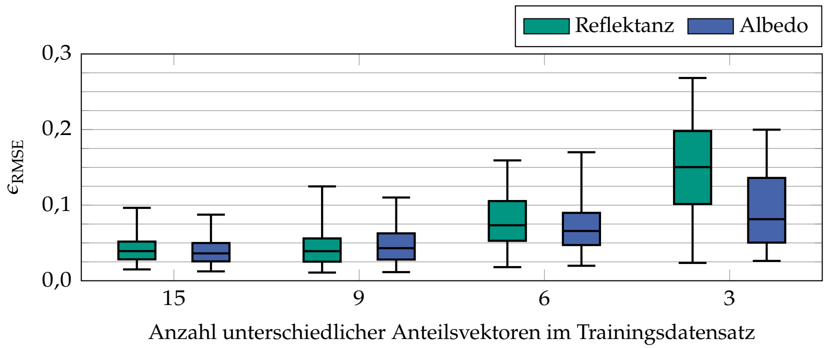


Abbildung 8.21 Darstellung von ϵ_{RMSE} als Box-Plots für den Datensatz $\mathcal{X}_{Q3}$. Vergleich Reflektanz und Albedo mit unterschiedlicher Anzahl an verschiedenen Anteilsvektoren im Trainingsdatensatz des generativen CNNs.

Im ersten Versuch werden alle 15 Anteilsvektoren genutzt. Hier sind die Spektren in der Albedo-Domäne leicht im Vorteil. Im zweiten werden alle Anteilsvektoren entfernt, die den Wert 0,75 enthalten, womit 9 Anteilsvektoren verbleiben. Hier sind die Ergebnisse insgesamt schlechter, der Unterschied zwischen den Domänen ist klein, bei Median und Box sind die Reflektanzspektren leicht im Vorteil, dafür gibt es größere Ausreißer. In einem weiteren Versuch werden zusätzlich alle Anteilsvektoren entfernt, die den Wert 0,25 enthalten, sodass nur noch 6 Anteilsvektoren verbleiben. Die Ergebnisse verschlechtern sich dadurch insgesamt noch weiter. Hier funktioniert die Entmischung in der Albedo-Domäne für einen großen Teil der Datenpunkte besser. Zuletzt werden auch die Datenpunkte entfernt, die den Wert 0,5 enthalten. Hier ist der Unterschied groß. Jedoch muss bedacht werden, dass jetzt nur noch die Reinspektren im Trainingsdatensatz enthalten sind. Damit kann das CNN die Mischcharakteristik des Datensatzes nicht lernen. Dies wird auch dadurch unterstrichen, dass damit nur ein kurzes Training möglich ist, bevor Überanpassung eintritt (siehe Tabelle 8.4). Dabei sind es in der Albedo-Domäne mehr Epochen als bei den Reflektanzspektren.

Dass die Anteilsschätzung mit den Spektren in der Albedo-Domäne trotzdem besser funktioniert als mit den Reflektanzspektren, liegt daran, dass im CNN lineare Operationen in den Faltungsschichten und den

Tabelle 8.4 Epochenzahl für das Training des CNNs für die spektrale Entmischung in Abhängigkeit der Anzahl unterschiedlicher Anteilsvektoren **a**.

	Anteilsvektoren			
Domäne	15	9	6	1
Reflektanz	81	91	121	11
Albedo	61	81	81	21

vollständig verbundenen Schichten durchgeführt werden. Die nichtlinearen Aktivierungsfunktionen haben ebenfalls einen Bereich, in dem sie näherungsweise linear sind. Somit ist es nicht verwunderlich, dass bei fehlenden Zwischenwerten im Trainingsdatensatz ein näherungsweise linearer Zusammenhang gelernt wird. Damit stellt die Nutzung der Albedo-Domäne bei besonders wenig verfügbaren Anteilsvektoren im Trainingsdatensatz tatsächlich einen Vorteil dar.

8.4 Zusammenfassung der Auswertung

Zusammenfassend lässt sich sagen, dass mit geeigneten Trainingsdaten die Verwendung von KNN zur spektralen Entmischung Vorteile mit sich bringt. Sind nur von den Reinstoffen echte Spektren verfügbar, muss der Trainingsdatensatz modellbasiert erstellt werden (siehe Abschnitt 8.3.3). Dies führt zu einem Vorteil gegenüber der korrespondierenden direkten Entmischung mit dem gleichen Mischmodell, wie es für die Datenerzeugung genutzt wird. Vor allem bei den nichtlinearen Mischmodellen FM, GBM und LQM ist dieser Vorteil, obwohl bei der Datenerzeugung nur ein fester Nichtlinearitätskoeffizient verwendet wird, groß, da diese schwierig zu optimieren sind. Der Vorteil wird geringer, je näher das verwendete Modell an der Realität ist. Dabei hat sich herausgestellt, dass die zufällige Reinspektrenwahl und die Modellierung als normalverteilte Zufallsvektoren bei ausreichend vorhandenen Spektren zu ähnlichen Ergebnissen führen.

Sind Trainingsdaten in Form von Mischspektren mit Anteilsvektor vorhanden, kann damit direkt ein CNN für die spektrale Entmischung trainiert werden. Dieses liefert bessere Ergebnisse als die modellbasier-

ten Methoden, wenn ausreichend Trainingsdaten zur Verfügung stehen ($\mathcal{X}_{Q3}$) oder die Modelle zu ungenau sind ($\mathcal{X}_{F4}$). Abgesehen von der spektralen Entmischung in der Albedo-Domäne werden auch beim Datensatz $\mathcal{X}_{Q4}$ durchweg bessere Ergebnisse mit dem CNN erzielt.

Die vorgestellten Verfahren zur Augmentierung verbessern die Ergebnisse des CNNs in allen Fällen, außer die Anteilsstufe, auf der die erzeugten Anteilsvektoren basieren, wird zu groß gewählt. Die Unterschiede zwischen den Verfahren und deren Varianten fallen gering aus. Es hängt außerdem vom Datensatz ab, welches Verfahren die größte Verbesserung bringt. Das GCNN liefert für alle untersuchten Datensätze gute Ergebnisse, besonders in der Variante, bei der die Kovarianzmatrix zur Regularisierung genutzt wird. Das RGAN kann die Streuung der Werte gegenüber dem GCNN noch verbessern. Dies ist jedoch nicht bei allen Datensätzen gelungen. Die Zuhilfenahme des Diskriminators bei der Datenerzeugung verändert die Ergebnisse nicht nennenswert, sodass darauf verzichtet werden kann. Das GPNN führt bei einem Datensatz ($\mathcal{X}_{Q3}$) ebenfalls zu guten Ergebnissen, jedoch kann es nicht mit dem GCNN mithalten, wenn weniger Trainingsdaten vorhanden sind. Der Vorteil beim GPNN ist das kleine, schnell zu trainierende KNN. Dabei hat sich herausgestellt, dass die Modellierung der Kovarianzfunktion kaum einen Unterschied macht. Dass die Modellierung der Spektrenvariabilität sinnvoll ist, ist an der Verbesserung der Ergebnisse zwischen GCNN und GCNN-Cov bzw. RGAN zu sehen. Dies deckt sich auch weitestgehend mit den Ergebnissen des direkten Vergleichs der erzeugten Daten mit den Testdaten in Abschnitt 8.2.3. Dabei ist zu beachten, dass bei den Verfahren GCNN-Cov und GPNN die Kovarianzmatrizen für die Anteilsvektoren explizit berechnet werden müssen. So ist deren Anwendung nur möglich, wenn entsprechend viele Spektren für jeden Anteilsvektor vorhanden sind. Bei allen anderen Verfahren zur Erzeugung von zusätzlichen Trainingsdaten ist dies nicht erforderlich.

Bei allen Augmentierungsverfahren spielt die Wahl der bei der Datenerzeugung zugrundeliegenden Anteilsstufe eine untergeordnete Rolle, solange diese nicht zu groß gewählt wird. Das liegt daran, dass die Netze zur Datenerzeugung keine zusätzliche Information zur Verfügung haben. Sie nutzen die vorhandenen Daten nur besser aus als das CNN für die spektrale Entmischung (siehe Kapitel 7). Damit ist auch hier kei-

ne beliebig feine Abstufung möglich. Zusätzliche Stützstellen bei den Anteilsvektoren unterstützen jedoch das CNN für die spektrale Entmischung beim Training. Zu beachten gilt auch, dass die zweitgrößte untersuchte Anteilsstufe ($\mathcal{S} = \frac{1}{8}$ bzw. $\mathcal{S} = \frac{1}{5}$) dazu führt, dass genau für die Anteilsstufen, die in den Testdatensätzen enthalten sind, Spektren erzeugt werden. Dies könnte dazu führen, dass diese Anteilstufen besonders gut abschneiden. Das spiegelt sich in den Ergebnissen jedoch nicht wider.

Die spektrale Entmischung mit Spektren in der Albedo-Domäne führt bei den modellbasierten Verfahren zu den besten Ergebnissen. Dies deckt sich mit der Theorie in Kapitel 2, da es sich bei allen Datensätzen um Mischungen feiner Pulver handelt. Bei den datenbasierten Verfahren gibt es kaum einen Unterschied zwischen der Verwendung von Reflektanzspektren und Spektren in der Albedo-Domäne. Das liegt daran, dass diese, auch bei wenig vorhandenen Trainingsdaten, den Zusammenhang implizit lernen (siehe Abschnitt 8.3.5).

9 Zusammenfassung und Ausblick

In der vorliegenden Arbeit wurden Verfahren zur spektralen Entmischung mit KNN vorgestellt. Dabei handelt es sich um Verfahren, die überwacht trainiert werden. Damit ist zum Training für jedes Spektrum ein zugehöriger Anteilsvektor im Datensatz erforderlich. Dies hat den Vorteil, dass die Zusammenhänge vollständig aus den Daten gelernt werden können und kein Modellwissen erforderlich ist. Nachteilig ist dabei, dass die dafür notwendigen Datensätze nur aufwendig zu erstellen sind. Bei den Daten wird von einem industriellen Umfeld ausgegangen, in dem die beteiligten Reinstoffe bekannt sind und es möglich ist, Stoffgemische zu erzeugen, von denen jeweils mehrere Aufnahmen gemacht werden können. Hierbei ist das Erstellen unterschiedlicher Aufnahmen weniger aufwendig als die Anfertigung der Stoffgemische. Für die Trainingsdatensätze werden systematisch Mischungen erzeugt. Dabei werden von jeder Mischung mehrere Aufnahmen erstellt, sodass daraus die statistischen Eigenschaften der Spektrenvariabilität bestimmt werden können.

In Kapitel 5 wurde ein CNN vorgestellt, das an die Anforderungen der spektralen Entmischung angepasst ist. Zum einen ist es im Vergleich zu gängigen KNN aus der Bildverarbeitung klein, sodass es mit wenig Trainingsdaten zurecht kommt. Zum anderen ist die Ausgangsschicht so ausgelegt, dass die Nebenbedingungen an die Anteilsvektoren stets erzwungen werden. Die Spektrenvariabilität wird dabei implizit über die Daten berücksichtigt. Hier wäre ein größeres KNN gegebenenfalls vorteilhaft, um möglichst invariant gegen Spektrenvariabilität zu werden. Dafür sind die Trainingsdaten allerdings nicht ausreichend. Es würde zu Überanpassung kommen und damit zu Problemen mit Anteilsvektoren führen, die zwischen den Anteilsvektoren des Trainingsdatensatzes liegen. Die kontinuierlichen mehrdimensionalen Ausgangsgrößen sind neben der Spektrenvariabilität eine große Herausforderung beim Training eines KNNs für die spektrale Entmischung, weil Trainingsdatensätze

immer nur eine endliche (und in der Praxis nicht beliebig große) Anzahl an Anteilsvektoren enthalten können.

Um dem entgegenzuwirken, wurden in Kapitel 7 Verfahren zur Datenaugmentierung vorgestellt. Dadurch stehen Trainingsdaten mit weiteren Anteilsvektoren, die zwischen den ursprünglichen Anteilsvektoren liegen, zur Verfügung. Bei den Verfahren zur Augmentierung handelt es sich um KNN, die in der Lage sind, die vorhandenen Trainingsdaten besser auszunutzen als das CNN für die spektrale Entmischung. Das geschieht beim GCNN durch Vertauschung von Ein- und Ausgangsdaten im Vergleich zum CNN für die spektrale Entmischung und zusätzlicher Einführung von zufälligen Eingangsgrößen. Beim RGAN wird das GCNN in eine GAN-Struktur integriert. Beim letzten Ansatz, dem GPNN, wird der Umweg über die Gauß-Prozesse gegangen, was in besonders einfachen KNN resultiert. Bei allen vorgestellten Verfahren wird die Spektrenvariabilität durch stochastische Komponenten modelliert. Es zeigt sich dabei in den meisten Parametrierungen eine Verbesserung gegenüber dem CNN für die spektrale Entmischung, das mit den ursprünglichen Trainingsdaten trainiert wird. Hierbei führt eine weitere Verkleinerung der Anteilsstufe zu keiner Verbesserung mehr, was daran liegt, dass keine realen Spektren für die zusätzlichen Mischverhältnisse hinzukommen. Stattdessen werden für neue Mischverhältnisse Spektren auf Basis der vorhandenen generiert, sodass schnell die Grenze der möglichen Verbesserung erreicht wird. Welches der Verfahren im konkreten Fall am besten funktioniert, hängt von den verwendeten Daten ab.

Die rein datenbasiert trainierten Verfahren sind den modellbasierten in den meisten Fällen überlegen. Bei den hier untersuchten Datensätzen ist es bei zwei von drei der Fall. Die modellbasierte spektrale Entmischung ist bei dem verbleibenden Datensatz nur dann im Vorteil, wenn eine genauere Modellierung der Mischung in der Albedo-Domäne genutzt wird. Daraus lässt sich ableiten, dass die datenbasierten Verfahren besser funktionieren können, wenn viele Trainingsdaten vorhanden sind oder eine ausreichend genaue Modellierung wegen unzureichendem Modellwissen nicht möglich ist.

Neben den datenbasierten Verfahren wurden in Kapitel 6 zwei modellbasierte Verfahren zur Datenerzeugung vorgestellt. Diese erzeugten Daten basieren auf den Reinspektren und dienen als Trainingsdaten für

das CNN für die spektrale Entmischung. Die Idee dabei ist, Vorteile einer spektralen Entmischung mit KNN zu nutzen, nämlich die Erzwingung der Nebenbedingungen durch die Netzstruktur und den Einbezug der Spektrenvariabilität durch die Trainingsdaten. Diese Verfahren übertreffen bei den meisten Datensätzen die direkte Anwendung desselben Mischmodells zur spektralen Entmischung. Dies gilt insbesondere für die untersuchten nichtlinearen Mischmodelle.

Wie bei datenbasierten Verfahren üblich, basieren die in Kapitel 8 vorgestellten Ergebnisse auf den verwendeten Datensätzen und sind daher nur für diese belegt. Bei den verwendeten Datensätzen handelt es sich um keine Spezialfälle, die besonders einfach spektral zu entmischen sind. Daher ist durchaus zu erwarten, dass für weitere Datensätze unter den beschriebenen Bedingungen mit den in dieser Arbeit vorgestellten Verfahren ähnliche Ergebnisse erzielt werden können. Die Hyperparameter der genutzten KNN müssen dazu angepasst werden. Auch der verwendete Wellenlängenbereich sollte idealerweise an die Anwendung angepasst werden.

Sollen die vorgestellten Verfahren zukünftig für spezielle (industrielle) Anwendungen eingesetzt werden, ist zu beachten, dass sie die Reflektanz nutzen. Daher lässt sich damit vor allem auf die Zusammensetzung der Oberfläche der zu untersuchenden Stoffgemische schließen. Deswegen eignen sich die Verfahren vor allem dann, wenn, wie hier, von einer lokalen Homogenität der Mischung ausgegangen werden kann. Alternativ kann, wenn das möglich ist, das zu untersuchende Stoffgemisch dünn unter der Kamera durchgeführt werden. Das wäre z. B. mit einem breiter werdenden Fließband realisierbar. Diese Einschränkung gilt aber für alle Verfahren, die die Reflektanz für die spektrale Entmischung nutzen.

Darüber hinaus eignen sich die erreichbaren Genauigkeiten nicht für jede Anwendung. Hier können die Ergebnisse dieser Arbeit einen Hinweis darauf geben, was möglich ist, vor allem, wenn es sich bei den zu untersuchenden Stoffgemischen um Pulvermischungen handelt. Es hat sich jedoch auch in dieser Arbeit beim Vergleich zwischen den Quarzsanden und den Farbpulvern gezeigt, dass die Ergebnisse stark datensatzabhängig und damit anwendungsabhängig sind. Diese Abhängigkeit gilt jedoch für alle untersuchten Verfahren.

Wie in dieser Arbeit gezeigt wird, können mit überwachten Trainingsverfahren bessere Ergebnisse als mit modellbasierten Verfahren erreicht werden. Vorraussetzung dafür ist die Verfügbarkeit entsprechender Trainingsdaten, was im Vergleich zwischen den beiden Datensätzen aus farbigen Quarzsanden deutlich wird. Damit der Aufwand der Datensatzerstellung nicht zu groß wird, dürfen nicht zu viele Reinstoffe enthalten sein, auch wenn durch die vorgestellten Augmentierungverfahren schon wenige reale Mischungen genügen. Sind bei einer Anwendung viele Reinstoffe beteiligt, bleiben die modellbasierten Verfahren aus Kapitel 6 eine Alternative, bei der je nach Mischmodell nur Aufnahmen der Reinstoffe oder zusätzlich wenige Mischspektren benötigt werden.

Auch methodisch lassen sich die vorgestellten Verfahren zukünftig erweitern. Für die datenbasierte Erzeugung zusätzlicher Trainingsdaten, wie sie in Kapitel 7 vorgestellt wird, gibt es eine Klasse an KNN, die in der Sprach- und Bildsynthese große Erfolge erzielen konnte [20, 97]. *Denoising Diffusion Probabilistic Models* [51, 87, 118] werden so trainiert, dass sie die Umkehrfunktion einer schrittweisen Verrauschung mit normalverteiltem Rauschen lernen. Dabei werden die Daten in vielen Schritten verrauscht, bis nur noch Rauschen übrig bleibt, was ebenfalls normalverteilt ist. Die trainierten Netze sind dann in der Lage, unbekannte Realisierungen von normalverteiltem Rauschen schrittweise zu „entrauschen“, wodurch neue Daten mit denselben Eigenschaften wie die Trainingsdaten generiert werden können. Wie beim *Conditional Variational Autoencoder* und beim CGAN existieren hier auch Ansätze, die eine Vorgabe von Bedingungen erlauben, was die Erzeugung von Datenpunkten einer bestimmten Klasse ermöglicht. Zukünftige Forschung könnte untersuchen, ob auch hier die Vorgabe kontinuierlicher Werte, die nicht im Trainingsdatensatz enthalten sind, möglich ist. Das ist bei der Erzeugung zusätzlicher Trainingsdaten für die spektrale Entmischung notwendig.

Anhang

A Herleitung: Stochastische Mischmodelle

In diesem Anhang werden Herleitungen zu Formeln ausführlich dargestellt, die den Rahmen der Hauptkapitel sprengen würden. Da er sich eher als Nachschlagewerk versteht, werden in diesem Anhang zum Teil Größen neu definiert und nicht alle Größen ins Symbolverzeichnis aufgenommen.

Die Beziehung (6.10) kann in wenigen Schritten mit Hilfe von (6.6) und (6.9) gezeigt werden:

$$\begin{aligned}
&\operatorname{Cov}\left\{\sum_{p=1}^{P} a_p\,\mathbf{m}_p, \sum_{p=1}^{P} a_p\,\mathbf{m}_p\right\} \\
&= \operatorname{E}\left\{\left(\sum_{p=1}^{P} a_p\,\mathbf{m}_p - \sum_{p=1}^{P} a_p\,\boldsymbol{\mu}_p\right)\left(\sum_{p=1}^{P} a_p\,\mathbf{m}_p - \sum_{p=1}^{P} a_p\,\boldsymbol{\mu}_p\right)^{\mathrm{T}}\right\} \\
&= \operatorname{E}\left\{\left(\sum_{p=1}^{P} a_p\left(\mathbf{m}_p - \boldsymbol{\mu}_p\right)\right)\left(\sum_{p=1}^{P} a_p\left(\mathbf{m}_p - \boldsymbol{\mu}_p\right)^{\mathrm{T}}\right)\right\} \\
&= \operatorname{E}\left\{\sum_{p=1}^{P}\sum_{q=1}^{P} a_p\left(\mathbf{m}_p - \boldsymbol{\mu}_p\right) a_q\left(\mathbf{m}_q - \boldsymbol{\mu}_q\right)^{\mathrm{T}}\right\} \\
&= \sum_{p=1}^{P}\sum_{q=1}^{P} a_p\,a_q\operatorname{E}\left\{\left(\mathbf{m}_p - \boldsymbol{\mu}_p\right)\left(\mathbf{m}_q - \boldsymbol{\mu}_q\right)^{\mathrm{T}}\right\} \\
&= \sum_{p=1}^{P}\sum_{q=1}^{P} a_p\,a_q\operatorname{Cov}\left\{\mathbf{m}_p, \mathbf{m}_q\right\}.
\end{aligned} \tag{A.1}$$

Der Zusammenhang (6.23) kann mit Hilfe von (6.9), (6.11), (6.13) und (6.14) hergeleitet werden:

$$\begin{aligned}
\boldsymbol{\Sigma}_{p\odot q,p\odot q} &= \mathrm{E}\Big\{\big(\mathbf{m}_p \odot \mathbf{m}_q - \boldsymbol{\mu}_p \odot \boldsymbol{\mu}_q\big)\big(\mathbf{m}_p \odot \mathbf{m}_q - \boldsymbol{\mu}_p \odot \boldsymbol{\mu}_q\big)^{\mathrm{T}}\Big\} \\
&= \mathrm{E}\Big\{\big(\mathbf{m}_p \odot \mathbf{m}_q\big)\big(\mathbf{m}_p \odot \mathbf{m}_q\big)^{\mathrm{T}} - \big(\mathbf{m}_p \odot \mathbf{m}_q\big)\big(\boldsymbol{\mu}_p \odot \boldsymbol{\mu}_q\big)^{\mathrm{T}} \\
&\quad - \big(\boldsymbol{\mu}_p \odot \boldsymbol{\mu}_q\big)\big(\mathbf{m}_p \odot \mathbf{m}_q\big)^{\mathrm{T}} + \big(\boldsymbol{\mu}_p \odot \boldsymbol{\mu}_q\big)\big(\boldsymbol{\mu}_p \odot \boldsymbol{\mu}_q\big)^{\mathrm{T}}\Big\} \\
&= \mathrm{E}\big\{\mathbf{m}_p\mathbf{m}_p^{\mathrm{T}}\big\} \odot \mathrm{E}\big\{\mathbf{m}_q\mathbf{m}_q^{\mathrm{T}}\big\} - \big(\boldsymbol{\mu}_p\boldsymbol{\mu}_p^{\mathrm{T}}\big) \odot \big(\boldsymbol{\mu}_q\boldsymbol{\mu}_q^{\mathrm{T}}\big) \\
&= \big(\boldsymbol{\Sigma}_{p,p} + \boldsymbol{\mu}_p\boldsymbol{\mu}_p^{\mathrm{T}}\big) \odot \big(\boldsymbol{\Sigma}_{q,q} + \boldsymbol{\mu}_q\boldsymbol{\mu}_q^{\mathrm{T}}\big) - \big(\boldsymbol{\mu}_p\boldsymbol{\mu}_p^{\mathrm{T}}\big) \odot \big(\boldsymbol{\mu}_q\boldsymbol{\mu}_q^{\mathrm{T}}\big) \\
&= \boldsymbol{\Sigma}_{p,p} \odot \boldsymbol{\Sigma}_{q,q} + \boldsymbol{\Sigma}_{p,p} \odot \big(\boldsymbol{\mu}_q\boldsymbol{\mu}_q^{\mathrm{T}}\big) + \boldsymbol{\Sigma}_{q,q} \odot \big(\boldsymbol{\mu}_p\boldsymbol{\mu}_p^{\mathrm{T}}\big) \,.
\end{aligned} \tag{A.2}$$

Der Zusammenhang (6.24) kann ebenfalls mit Hilfe von (6.9), (6.11), (6.13) und (6.14) hergeleitet werden, wobei **1** ein Vektor der passenden Länge bestehend aus Einsen ist:

$$\begin{aligned}
\boldsymbol{\Sigma}_{p,p\odot q} &= \mathrm{E}\Big\{\big(\mathbf{m}_p - \boldsymbol{\mu}_p\big)\big(\mathbf{m}_p \odot \mathbf{m}_q - \boldsymbol{\mu}_p \odot \boldsymbol{\mu}_q\big)^{\mathrm{T}}\Big\} \\
&= \mathrm{E}\Big\{\mathbf{m}_p\big(\mathbf{m}_p \odot \mathbf{m}_q\big)^{\mathrm{T}} - \mathbf{m}_p\big(\boldsymbol{\mu}_p \odot \boldsymbol{\mu}_q\big)^{\mathrm{T}} \\
&\quad - \boldsymbol{\mu}_p\big(\mathbf{m}_p \odot \mathbf{m}_q\big)^{\mathrm{T}} + \boldsymbol{\mu}_p\big(\boldsymbol{\mu}_p \odot \boldsymbol{\mu}_q\big)^{\mathrm{T}}\Big\} \\
&= \mathrm{E}\big\{\mathbf{m}_p\mathbf{m}_p^{\mathrm{T}}\big\} \odot \mathrm{E}\big\{\mathbf{1}\,\mathbf{m}_q^{\mathrm{T}}\big\} - \big(\boldsymbol{\mu}_p\boldsymbol{\mu}_p^{\mathrm{T}}\big) \odot \big(\mathbf{1}\,\boldsymbol{\mu}_q^{\mathrm{T}}\big) \\
&= \big(\boldsymbol{\Sigma}_{p,p} + \boldsymbol{\mu}_p\boldsymbol{\mu}_p^{\mathrm{T}}\big) \odot \big(\mathbf{1}\,\boldsymbol{\mu}_q^{\mathrm{T}}\big) - \big(\boldsymbol{\mu}_p\boldsymbol{\mu}_p^{\mathrm{T}}\big) \odot \big(\mathbf{1}\,\boldsymbol{\mu}_q^{\mathrm{T}}\big) \\
&= \boldsymbol{\Sigma}_{p,p} \odot \big(\mathbf{1}\,\boldsymbol{\mu}_q^{\mathrm{T}}\big) \,.
\end{aligned} \tag{A.3}$$

Dabei genügt es, den ersten Summanden herzuleiten, da der zweite Summand aufgrund von (6.7) dem transponierten ersten Summanden entspricht.

Auch der Zusammenhang (6.25) kann mit Hilfe von (6.9), (6.11), (6.13) und (6.14) hergeleitet werden:

$$
\begin{aligned}
\boldsymbol{\Sigma}_{p\odot q,p\odot r} &= \mathrm{E}\Big\{\left(\mathbf{m}_p \odot \mathbf{m}_q - \boldsymbol{\mu}_p \odot \boldsymbol{\mu}_q\right)\left(\mathbf{m}_p \odot \mathbf{m}_r - \boldsymbol{\mu}_p \odot \boldsymbol{\mu}_r\right)^{\mathrm{T}}\Big\} \\
&= \mathrm{E}\Big\{\left(\mathbf{m}_p \odot \mathbf{m}_q\right)\left(\mathbf{m}_p \odot \mathbf{m}_r\right)^{\mathrm{T}} - \left(\mathbf{m}_p \odot \mathbf{m}_q\right)\left(\boldsymbol{\mu}_p \odot \boldsymbol{\mu}_r\right)^{\mathrm{T}} \\
&\quad - \left(\boldsymbol{\mu}_p \odot \boldsymbol{\mu}_q\right)\left(\mathbf{m}_p \odot \mathbf{m}_r\right)^{\mathrm{T}} + \left(\boldsymbol{\mu}_p \odot \boldsymbol{\mu}_q\right)\left(\boldsymbol{\mu}_p \odot \boldsymbol{\mu}_r\right)^{\mathrm{T}}\Big\} \\
&= \mathrm{E}\Big\{\mathbf{m}_p\mathbf{m}_p^{\mathrm{T}}\Big\} \odot \left(\mathrm{E}\Big\{\mathbf{m}_q\Big\}\,\mathrm{E}\{\mathbf{m}_r\}^{\mathrm{T}}\right) - \left(\boldsymbol{\mu}_p\boldsymbol{\mu}_p^{\mathrm{T}}\right) \odot \left(\boldsymbol{\mu}_q\boldsymbol{\mu}_r^{\mathrm{T}}\right) \\
&= \left(\boldsymbol{\Sigma}_{p,p} + \boldsymbol{\mu}_p\boldsymbol{\mu}_p^{\mathrm{T}}\right) \odot \left(\boldsymbol{\mu}_q\boldsymbol{\mu}_r^{\mathrm{T}}\right) - \left(\boldsymbol{\mu}_p\boldsymbol{\mu}_p^{\mathrm{T}}\right) \odot \left(\boldsymbol{\mu}_q\boldsymbol{\mu}_r^{\mathrm{T}}\right) \\
&= \boldsymbol{\Sigma}_{p,p} \odot \left(\boldsymbol{\mu}_q\boldsymbol{\mu}_r^{\mathrm{T}}\right) .
\end{aligned}
\tag{A.4}
$$

Dabei genügt es, den ersten Summanden herzuleiten, da der zweite Summand aufgrund von (6.7) dem transponierten ersten Summanden entspricht.

Die Zusammenhänge (6.30) und (6.31) können gemeinsam hergeleitet werden, da (6.30) ein Spezialfall von (6.31) mit $\mathbf{m}_q = \mathbf{1}$ ist (vgl. (A.3) und (A.4), wo dies für das GBM ausführlich durchgeführt wird). Auch hier genügt es, den ersten Summanden herzuleiten, da der zweite Summand aufgrund von (6.7) dem transponierten ersten Summanden entspricht. Der Zusammenhang (6.31) kann mit Hilfe von (6.9), (6.11), (6.13), (6.14), (6.15) und (6.16) hergeleitet werden. Dazu wird zunächst für einen normalverteilten Zufallsvektor $\mathbf{m} = \mathbf{m}_p$ mit $\tilde{\mathbf{m}} = \mathbf{m} - \boldsymbol{\mu}_{\mathbf{m}}$ der folgende Zusammenhang benötigt:

$$
\begin{aligned}
\mathrm{E}\Big\{(\mathbf{m} \odot \mathbf{m})\,\mathbf{m}^{\mathrm{T}}\Big\} &= \mathrm{E}\Big\{((\tilde{\mathbf{m}} + \boldsymbol{\mu}_{\mathbf{m}}) \odot (\tilde{\mathbf{m}} + \boldsymbol{\mu}_{\mathbf{m}}))\,(\tilde{\mathbf{m}} + \boldsymbol{\mu}_{\mathbf{m}})^{\mathrm{T}}\Big\} \\
&= \mathrm{E}\Big\{(\tilde{\mathbf{m}} \odot \tilde{\mathbf{m}} + 2\,\tilde{\mathbf{m}} \odot \boldsymbol{\mu}_{\mathbf{m}} + \boldsymbol{\mu}_{\mathbf{m}} \odot \boldsymbol{\mu}_{\mathbf{m}})\,(\tilde{\mathbf{m}} + \boldsymbol{\mu}_{\mathbf{m}})^{\mathrm{T}}\Big\} \\
&= \mathrm{E}\Big\{(\tilde{\mathbf{m}} \odot \tilde{\mathbf{m}})\,\tilde{\mathbf{m}}^{\mathrm{T}} + (\tilde{\mathbf{m}} \odot \tilde{\mathbf{m}})\,\boldsymbol{\mu}_{\mathbf{m}}^{\mathrm{T}} + 2\,(\tilde{\mathbf{m}} \odot \boldsymbol{\mu}_{\mathbf{m}})\,\tilde{\mathbf{m}}^{\mathrm{T}} + 2\,(\tilde{\mathbf{m}} \odot \boldsymbol{\mu}_{\mathbf{m}})\,\boldsymbol{\mu}_{\mathbf{m}}^{\mathrm{T}} \\
&\quad + (\boldsymbol{\mu}_{\mathbf{m}} \odot \boldsymbol{\mu}_{\mathbf{m}})\,\tilde{\mathbf{m}}^{\mathrm{T}} + (\boldsymbol{\mu}_{\mathbf{m}} \odot \boldsymbol{\mu}_{\mathbf{m}})\,\boldsymbol{\mu}_{\mathbf{m}}^{\mathrm{T}}\Big\} \\
&= \mathrm{E}\Big\{(\tilde{\mathbf{m}} \odot \tilde{\mathbf{m}})\,\tilde{\mathbf{m}}^{\mathrm{T}}\Big\} + \mathrm{E}\{\tilde{\mathbf{m}} \odot \tilde{\mathbf{m}}\}\,\boldsymbol{\mu}_{\mathbf{m}}^{\mathrm{T}} + 2\,\mathrm{E}\Big\{\tilde{\mathbf{m}}\tilde{\mathbf{m}}^{\mathrm{T}}\Big\} \odot (\boldsymbol{\mu}_{\mathbf{m}}\,\mathbf{1})^{\mathrm{T}} \\
&\quad + 2\,(\mathrm{E}\{\tilde{\mathbf{m}}\} \odot \boldsymbol{\mu}_{\mathbf{m}})\,\boldsymbol{\mu}_{\mathbf{m}}^{\mathrm{T}} + (\boldsymbol{\mu}_{\mathbf{m}} \odot \boldsymbol{\mu}_{\mathbf{m}})\,\mathrm{E}\{\tilde{\mathbf{m}}\}^{\mathrm{T}} + (\boldsymbol{\mu}_{\mathbf{m}} \odot \boldsymbol{\mu}_{\mathbf{m}})\,\boldsymbol{\mu}_{\mathbf{m}}^{\mathrm{T}} \\
&= \mathrm{E}\{\tilde{\mathbf{m}} \odot \tilde{\mathbf{m}}\}\,\boldsymbol{\mu}_{\mathbf{m}}^{\mathrm{T}} + 2\,\mathrm{E}\Big\{\tilde{\mathbf{m}}\tilde{\mathbf{m}}^{\mathrm{T}}\Big\} \odot (\boldsymbol{\mu}_{\mathbf{m}}\,\mathbf{1})^{\mathrm{T}} + (\boldsymbol{\mu}_{\mathbf{m}} \odot \boldsymbol{\mu}_{\mathbf{m}})\,\boldsymbol{\mu}_{\mathbf{m}}^{\mathrm{T}} \\
&= d\,(\boldsymbol{\Sigma}_{\mathbf{m},\mathbf{m}})\,\boldsymbol{\mu}_{\mathbf{m}}^{\mathrm{T}} + 2\,\boldsymbol{\Sigma}_{\mathbf{m},\mathbf{m}} \odot \left(\boldsymbol{\mu}_{\mathbf{m}}\,\mathbf{1}^{\mathrm{T}}\right) + (\boldsymbol{\mu}_{\mathbf{m}} \odot \boldsymbol{\mu}_{\mathbf{m}})\,\boldsymbol{\mu}_{\mathbf{m}}^{\mathrm{T}} .
\end{aligned}
\tag{A.5}
$$

Die Herleitung des Zusammenhangs (6.31) ist schließlich:

$$
\begin{aligned}
\boldsymbol{\Sigma}_{p\odot p,p\odot q} &= \mathrm{E}\left\{\left(\mathbf{m}_p \odot \mathbf{m}_p\right)\left(\mathbf{m}_p \odot \mathbf{m}_q\right)^{\mathrm{T}}\right\} - \mathrm{E}\left\{\mathbf{m}_p \odot \mathbf{m}_p\right\}\left(\boldsymbol{\mu}_p \odot \boldsymbol{\mu}_q\right)^{\mathrm{T}} \\
&= \mathrm{E}\left\{\left(\mathbf{m}_p \odot \mathbf{m}_p\right)\mathbf{m}_p^{\mathrm{T}}\right\} \odot \left(\mathbf{1}\,\boldsymbol{\mu}_q^{\mathrm{T}}\right) - \left(\boldsymbol{\mu}_p \odot \boldsymbol{\mu}_p\right)\left(\boldsymbol{\mu}_p \odot \boldsymbol{\mu}_q\right)^{\mathrm{T}} \\
&\quad - d\left(\boldsymbol{\Sigma}_{p,p}\right)\left(\boldsymbol{\mu}_p \odot \boldsymbol{\mu}_q\right)^{\mathrm{T}} \\
&= d\left(\boldsymbol{\Sigma}_{p,p}\right)\left(\boldsymbol{\mu}_p \odot \boldsymbol{\mu}_q\right)^{\mathrm{T}} + 2\,\boldsymbol{\Sigma}_{p,p} \odot \left(\boldsymbol{\mu}_p\boldsymbol{\mu}_q^{\mathrm{T}}\right) + \left(\boldsymbol{\mu}_p \odot \boldsymbol{\mu}_p\right)\left(\boldsymbol{\mu}_p \odot \boldsymbol{\mu}_q\right)^{\mathrm{T}} \\
&\quad - \left(\boldsymbol{\mu}_p \odot \boldsymbol{\mu}_p\right)\left(\boldsymbol{\mu}_p \odot \boldsymbol{\mu}_q\right)^{\mathrm{T}} - d\left(\boldsymbol{\Sigma}_{p,p}\right)\left(\boldsymbol{\mu}_p \odot \boldsymbol{\mu}_q\right)^{\mathrm{T}} \\
&= 2\,\boldsymbol{\Sigma}_{p,p} \odot \left(\boldsymbol{\mu}_p\boldsymbol{\mu}_q^{\mathrm{T}}\right)\ .
\end{aligned}
\tag{A.6}
$$

Der Zusammenhang (6.32) kann mit Hilfe von (6.9), (6.11), (6.13), (6.14), (6.15) und (6.16) hergeleitet werden. Dazu wird zunächst für einen normalverteilten Zufallsvektor $\mathbf{m} = \mathbf{m}_p$ mit $\tilde{\mathbf{m}} = \mathbf{m} - \boldsymbol{\mu}_{\mathbf{m}}$ der folgende Zusammenhang benötigt (zu Gunsten der Lesbarkeit gilt hier $\boldsymbol{\mu} = \boldsymbol{\mu}_{\mathbf{m}}$):

$$
\begin{aligned}
&\mathrm{E}\left\{(\mathbf{m} \odot \mathbf{m})\,(\mathbf{m} \odot \mathbf{m})^{\mathrm{T}}\right\} \\
&= \mathrm{E}\left\{((\tilde{\mathbf{m}} + \boldsymbol{\mu}) \odot (\tilde{\mathbf{m}} + \boldsymbol{\mu}))\,((\tilde{\mathbf{m}} + \boldsymbol{\mu}) \odot (\tilde{\mathbf{m}} + \boldsymbol{\mu}))^{\mathrm{T}}\right\} \\
&= \mathrm{E}\left\{(\tilde{\mathbf{m}} \odot \tilde{\mathbf{m}} + 2\,\tilde{\mathbf{m}} \odot \boldsymbol{\mu} + \boldsymbol{\mu} \odot \boldsymbol{\mu})\,(\tilde{\mathbf{m}} \odot \tilde{\mathbf{m}} + 2\,\tilde{\mathbf{m}} \odot \boldsymbol{\mu} + \boldsymbol{\mu} \odot \boldsymbol{\mu})^{\mathrm{T}}\right\} \\
&= \mathrm{E}\Big\{(\tilde{\mathbf{m}} \odot \tilde{\mathbf{m}})\,(\tilde{\mathbf{m}} \odot \tilde{\mathbf{m}})^{\mathrm{T}} + 2\,(\tilde{\mathbf{m}} \odot \tilde{\mathbf{m}})\,(\tilde{\mathbf{m}} \odot \boldsymbol{\mu})^{\mathrm{T}} + (\tilde{\mathbf{m}} \odot \tilde{\mathbf{m}})\,(\boldsymbol{\mu} \odot \boldsymbol{\mu})^{\mathrm{T}} \\
&\quad + 2\,(\tilde{\mathbf{m}} \odot \boldsymbol{\mu})\,(\tilde{\mathbf{m}} \odot \tilde{\mathbf{m}})^{\mathrm{T}} + 4\,(\tilde{\mathbf{m}} \odot \boldsymbol{\mu})\,(\tilde{\mathbf{m}} \odot \boldsymbol{\mu})^{\mathrm{T}} + 2\,(\tilde{\mathbf{m}} \odot \boldsymbol{\mu})\,(\boldsymbol{\mu} \odot \boldsymbol{\mu})^{\mathrm{T}} \\
&\quad + (\boldsymbol{\mu} \odot \boldsymbol{\mu})\,(\tilde{\mathbf{m}} \odot \tilde{\mathbf{m}})^{\mathrm{T}} + 2\,(\boldsymbol{\mu} \odot \boldsymbol{\mu})\,(\tilde{\mathbf{m}} \odot \boldsymbol{\mu})^{\mathrm{T}} + (\boldsymbol{\mu} \odot \boldsymbol{\mu})\,(\boldsymbol{\mu} \odot \boldsymbol{\mu})^{\mathrm{T}}\Big\} \\
&= \mathrm{E}\left\{(\tilde{\mathbf{m}} \odot \tilde{\mathbf{m}})\,(\tilde{\mathbf{m}} \odot \tilde{\mathbf{m}})^{\mathrm{T}}\right\} + 2\,\mathrm{E}\left\{(\tilde{\mathbf{m}} \odot \tilde{\mathbf{m}})\,\tilde{\mathbf{m}}^{\mathrm{T}}\right\} \odot \left(\mathbf{1}\,\boldsymbol{\mu}^{\mathrm{T}}\right) \\
&\quad + \mathrm{E}\{\tilde{\mathbf{m}} \odot \tilde{\mathbf{m}}\}\,(\boldsymbol{\mu} \odot \boldsymbol{\mu})^{\mathrm{T}} + 2\,\mathrm{E}\left\{\tilde{\mathbf{m}}\,(\tilde{\mathbf{m}} \odot \tilde{\mathbf{m}})^{\mathrm{T}}\right\} \odot \left(\boldsymbol{\mu}\,\mathbf{1}^{\mathrm{T}}\right) \\
&\quad + 4\,\mathrm{E}\left\{\tilde{\mathbf{m}}\tilde{\mathbf{m}}^{\mathrm{T}}\right\} \odot \left(\boldsymbol{\mu}\boldsymbol{\mu}^{\mathrm{T}}\right) + 2\left(\mathrm{E}\{\tilde{\mathbf{m}}\}\,\mathbf{1}^{\mathrm{T}}\right) \odot \left(\boldsymbol{\mu}\,(\boldsymbol{\mu} \odot \boldsymbol{\mu})^{\mathrm{T}}\right) \\
&\quad + (\boldsymbol{\mu} \odot \boldsymbol{\mu})\,\mathrm{E}\{\tilde{\mathbf{m}} \odot \tilde{\mathbf{m}}\}^{\mathrm{T}} + 2\left((\boldsymbol{\mu} \odot \boldsymbol{\mu})\,\boldsymbol{\mu}^{\mathrm{T}}\right) \odot \left(\mathbf{1}\,\mathrm{E}\{\tilde{\mathbf{m}}\}^{\mathrm{T}}\right) \\
&\quad + (\boldsymbol{\mu} \odot \boldsymbol{\mu})\,(\boldsymbol{\mu} \odot \boldsymbol{\mu})^{\mathrm{T}} \\
&= d\left(\boldsymbol{\Sigma}_{\mathbf{m},\mathbf{m}}\right) d\left(\boldsymbol{\Sigma}_{\mathbf{m},\mathbf{m}}\right)^{\mathrm{T}} + 2\,\boldsymbol{\Sigma}_{\mathbf{m},\mathbf{m}} \odot \boldsymbol{\Sigma}_{\mathbf{m},\mathbf{m}} + d\left(\boldsymbol{\Sigma}_{\mathbf{m},\mathbf{m}}\right)(\boldsymbol{\mu} \odot \boldsymbol{\mu})^{\mathrm{T}} \\
&\quad + 4\,\boldsymbol{\Sigma}_{\mathbf{m},\mathbf{m}} \odot \left(\boldsymbol{\mu}\boldsymbol{\mu}^{\mathrm{T}}\right) + (\boldsymbol{\mu} \odot \boldsymbol{\mu})\,d\left(\boldsymbol{\Sigma}_{\mathbf{m},\mathbf{m}}\right)^{\mathrm{T}} + (\boldsymbol{\mu} \odot \boldsymbol{\mu})\,(\boldsymbol{\mu} \odot \boldsymbol{\mu})^{\mathrm{T}}\ .
\end{aligned}
\tag{A.7}
$$

Die Herleitung des Zusammenhangs (6.32) ist schließlich (zu Gunsten der Lesbarkeit gelten hier $\mathbf{m} = \mathbf{m}_p$, $\boldsymbol{\mu} = \boldsymbol{\mu}_p$, und $\boldsymbol{\Sigma} = \boldsymbol{\Sigma}_{p,p}$):

$$\begin{aligned}
\boldsymbol{\Sigma}_{p\odot p,p\odot p} &= \mathrm{E}\Big\{(\mathbf{m}\odot\mathbf{m})\,(\mathbf{m}\odot\mathbf{m})^{\mathrm{T}}\Big\} - \mathrm{E}\{\mathbf{m}\odot\mathbf{m}\}\,\mathrm{E}\{\mathbf{m}\odot\mathbf{m}\}^{\mathrm{T}} \\
&= \mathrm{E}\Big\{(\mathbf{m}\odot\mathbf{m})\,(\mathbf{m}\odot\mathbf{m})^{\mathrm{T}}\Big\} - (\boldsymbol{\mu}\odot\boldsymbol{\mu} + d(\boldsymbol{\Sigma}))\,(\boldsymbol{\mu}\odot\boldsymbol{\mu} + d(\boldsymbol{\Sigma}))^{\mathrm{T}} \\
&= d(\boldsymbol{\Sigma})\,d(\boldsymbol{\Sigma})^{\mathrm{T}} + 2\,\boldsymbol{\Sigma}\odot\boldsymbol{\Sigma} + d(\boldsymbol{\Sigma})\,(\boldsymbol{\mu}\odot\boldsymbol{\mu})^{\mathrm{T}} + 4\,\boldsymbol{\Sigma}\odot\left(\boldsymbol{\mu}\boldsymbol{\mu}^{\mathrm{T}}\right) \\
&\quad + (\boldsymbol{\mu}\odot\boldsymbol{\mu})\,d(\boldsymbol{\Sigma})^{\mathrm{T}} + (\boldsymbol{\mu}\odot\boldsymbol{\mu})\,(\boldsymbol{\mu}\odot\boldsymbol{\mu})^{\mathrm{T}} - (\boldsymbol{\mu}\odot\boldsymbol{\mu})\,(\boldsymbol{\mu}\odot\boldsymbol{\mu})^{\mathrm{T}} \\
&\quad - (\boldsymbol{\mu}\odot\boldsymbol{\mu})\,d(\boldsymbol{\Sigma}) - d(\boldsymbol{\Sigma})\,(\boldsymbol{\mu}\odot\boldsymbol{\mu})^{\mathrm{T}} - d(\boldsymbol{\Sigma})\,d(\boldsymbol{\Sigma})^{\mathrm{T}} \\
&= 2\,\boldsymbol{\Sigma}\odot\boldsymbol{\Sigma} + 4\,\boldsymbol{\Sigma}\odot\left(\boldsymbol{\mu}\boldsymbol{\mu}^{\mathrm{T}}\right)\,.
\end{aligned} \tag{A.8}$$

B Weitere Ergebnisse

Dieser Anhang enthält zusätzliche Ergebnisse bzw. Ergebnisse in einer anderen Darstellungsform als der Hauptteil dieser Arbeit. Dabei versteht sich dieser Anhang als Nachschlagewerk, auf das an den entsprechenden Stellen des Hauptteils verwiesen wird. Im Abschnitt B.1 werden die Ergebnisse der spektralen Entmischung für jeweils eine Anteilsstufe dargestellt. Dabei werden in je einem Schaubild die Box-Plots aller Verfahren der Datenerzeugung bzw. aller Verfahren der Datenerweiterung dargestellt. In Abschnitt B.2 folgt eine Darstellung des Gütemaßes ϵ_{RMSE} ausgewählter Verfahren in Form von Histogrammen, sodass ein genauerer Eindruck über die Verteilung gewonnen werden kann. Danach in Abschnitt B.3 sind die Nichtlinearitätskoeffizienten aufgelistet, die sich für die modellbasierte Datenerzeugung mit dem GBM und dem LQM ergeben haben. Zuletzt in Abschnitt B.4 folgen Beispielspektren, die mit den vorgestellten Verfahren erzeugt worden sind.

B.1 Ergebnisse der spektralen Entmischung im Vergleich

In diesem Abschnitt werden die Ergebnisse der spektralen Entmischung aus Abschnitt 8.3 zur besseren Vergleichbarkeit für eine Anteilsstufe je Datensatz dargestellt. Dabei wird immer die zweitkleinste getestete Anteilsstufe genutzt, da bei dieser keine nennenswerten Ausreißer auftreten.

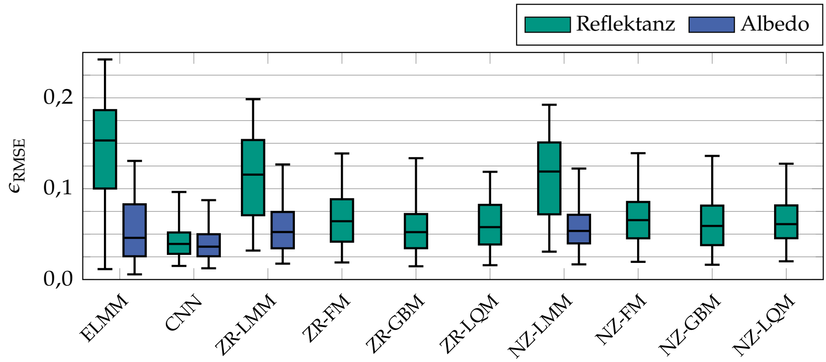


Abbildung B.1 Darstellung von ϵ_{RMSE} als Box-Plots für den Datensatz $\mathcal{X}_{\text{Q3}}$. Datenerzeugung mit Anteilsstufe $\mathcal{S} = \frac{1}{10}$.

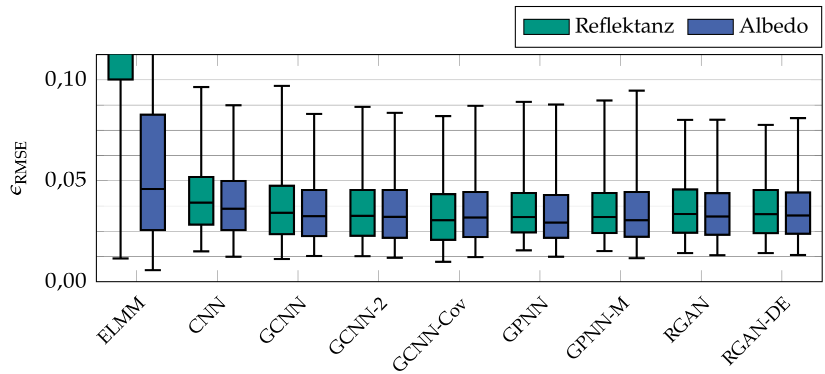


Abbildung B.2 Darstellung von ϵ_{RMSE} als Box-Plots für den Datensatz $\mathcal{X}_{\text{Q3}}$. Datenaugmentierung mit Anteilsstufe $\mathcal{S} = \frac{1}{10}$. Box-Plots für ELMM sind vollständig in Abbildung B.1 zu sehen und dienen hier nur der Orientierung.

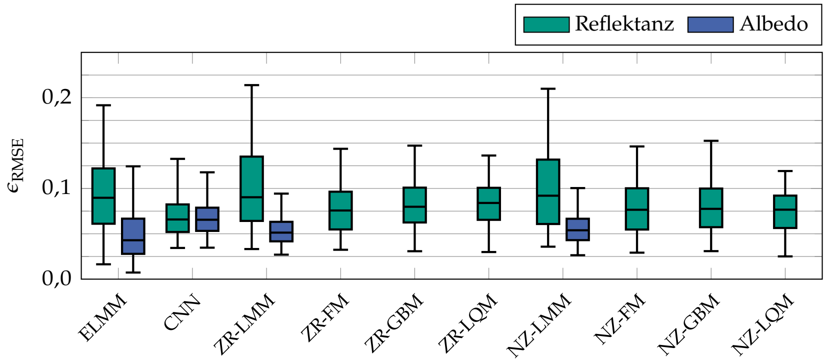


Abbildung B.3 Darstellung von ϵ_{RMSE} als Box-Plots für den Datensatz $\mathcal{X}_{\text{Q4}}$. Datenerzeugung mit Anteilsstufe $\mathcal{S} = \frac{1}{8}$.

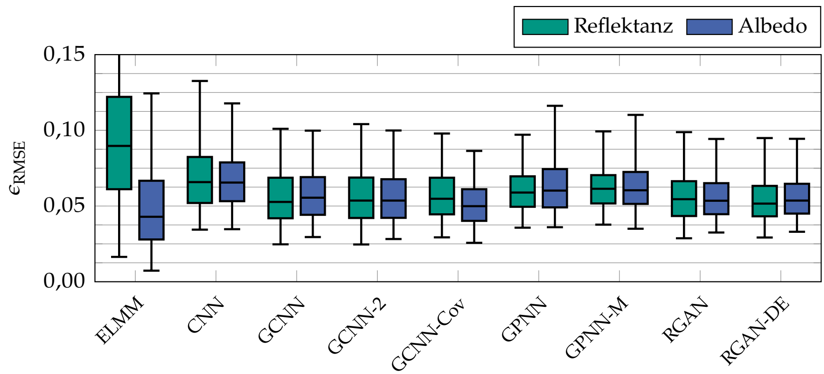


Abbildung B.4 Darstellung von ϵ_{RMSE} als Box-Plots für den Datensatz $\mathcal{X}_{\text{Q4}}$. Datenaugmentierung mit Anteilsstufe $\mathcal{S} = \frac{1}{8}$. Box-Plots für ELMM sind vollständig in Abbildung B.3 zu sehen und dienen hier nur der Orientierung.

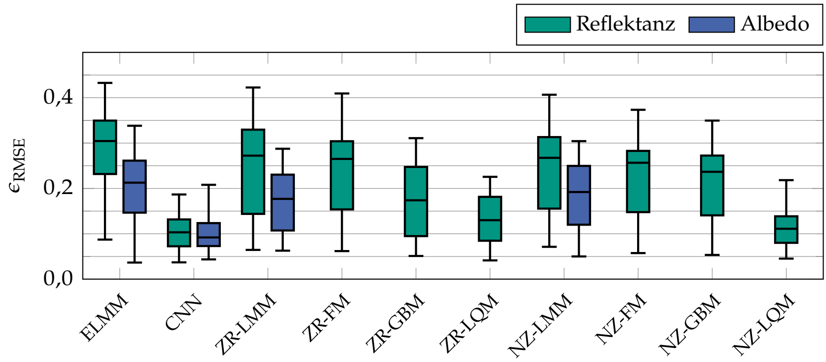


Abbildung B.5 Darstellung von ϵ_{RMSE} als Box-Plots für den Datensatz $\mathcal{X}_{F4}$. Datenerzeugung mit Anteilsstufe $\mathcal{S} = \frac{1}{8}$.

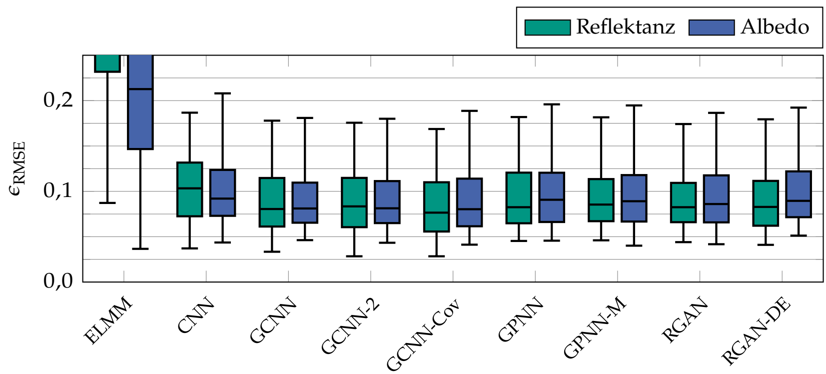


Abbildung B.6 Darstellung von ϵ_{RMSE} als Box-Plots für den Datensatz $\mathcal{X}_{F4}$. Datenaugmentierung mit Anteilsstufe $\mathcal{S} = \frac{1}{8}$. Box-Plots für ELMM sind vollständig in Abbildung B.5 zu sehen und dienen hier nur der Orientierung.

B.2 Histogramme der spektralen Entmischung

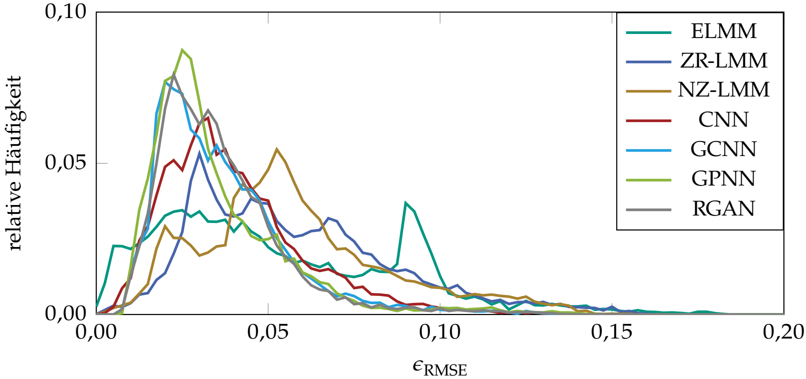


Abbildung B.7 Darstellung von ϵ_{RMSE} als Histogramm mit 81 Intervallen für den Datensatz $\mathcal{X}_{\text{Q3}}$. Vergleich von Datenaugmentierung und Erzeugung in der Albedo-Domäne für Schrittweite $\mathcal{S} = \frac{1}{10}$. Zur besseren Vergleichbarkeit sind die Histogramme als Linien dargestellt.

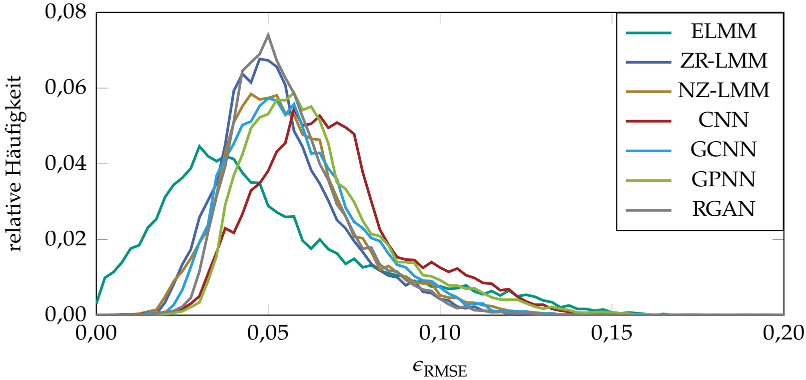


Abbildung B.8 Darstellung von ϵ_{RMSE} als Histogramm mit 81 Intervallen für den Datensatz $\mathcal{X}_{\text{Q4}}$. Vergleich von Datenaugmentierung und Erzeugung in der Albedo-Domäne für Schrittweite $\mathcal{S} = \frac{1}{8}$. Zur besseren Vergleichbarkeit sind die Histogramme als Linien dargestellt.

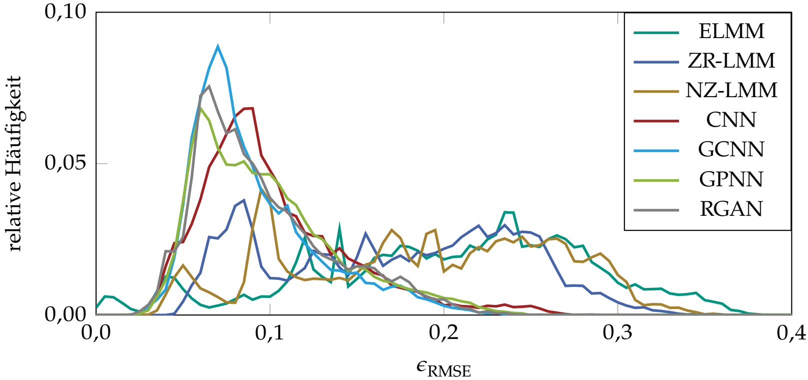


Abbildung B.9 Darstellung von ϵ_{RMSE} als Histogramm mit 81 Intervallen für den Datensatz $\mathcal{X}_{\mathrm{F4}}$. Vergleich von Datenaugmentierung und Erzeugung in der Albedo-Domäne für Schrittweite $\mathcal{S} = \frac{1}{8}$. Zur besseren Vergleichbarkeit sind die Histogramme als Linien dargestellt.

B.3 Nichtlinearitätskoeffizienten

Tabelle B.1 Nichtlinearitätskoeffizienten γ (GBM) bzw. δ (LQM) für den Datensatz $\mathcal{X}_{\mathrm{Q3}}$.

Verfahren	ermittelt bei	Anteilsstufe $\frac{1}{5}$	$\frac{1}{8}$	$\frac{1}{10}$	$\frac{1}{20}$
ZR-GBM	Validierung	1,35	1,55	1,45	1,60
	Test	1,25	1,55	1,30	1,70
NZ-GBM	Validierung	1,75	1,30	1,25	2,05
	Test	1,75	1,30	1,45	1,55
ZR-LQM	Validierung	0,040	0,054	0,050	0,044
	Test	0,040	0,046	0,040	0,044
NZ-LQM	Validierung	0,038	0,042	0,038	0,048
	Test	0,038	0,042	0,038	0,048

Tabelle B.2 Nichtlinearitätskoeffizienten γ (GBM) bzw. δ (LQM) für den Datensatz $\mathcal{X}_{Q4}$.

Verfahren	ermittelt bei	Anteilsstufe $\frac{1}{4}$	$\frac{1}{5}$	$\frac{1}{8}$	$\frac{1}{16}$
ZR-GBM	Validierung	2,65	1,45	1,95	2,25
	Test	0,85	1,40	1,20	1,05
NZ-GBM	Validierung	2,20	2,65	2,50	1,50
	Test	0,85	1,70	0,85	1,15
ZR-LQM	Validierung	0,018	0,026	0,024	0,020
	Test	0,018	0,012	0,016	0,020
NZ-LQM	Validierung	0,030	0,026	0,024	0,024
	Test	0,018	0,026	0,024	0,018

Tabelle B.3 Nichtlinearitätskoeffizienten γ (GBM) bzw. δ (LQM) für den Datensatz $\mathcal{X}_{F4}$.

Verfahren	ermittelt bei	Anteilsstufe $\frac{1}{4}$	$\frac{1}{5}$	$\frac{1}{8}$	$\frac{1}{16}$
ZR-GBM	Validierung	2,55	2,70	2,30	2,55
	Test	2,55	2,70	2,30	2,55
NZ-GBM	Validierung	0,20	0,95	1,75	0,95
	Test	2,65	2,85	2,80	2,95
ZR-LQM	Validierung	0,074	0,072	0,080	0,062
	Test	0,074	0,072	0,066	0,062
NZ-LQM	Validierung	0,074	0,080	0,076	0,068
	Test	0,074	0,080	0,076	0,068

B.4 Beispiele für erzeugte Spektren

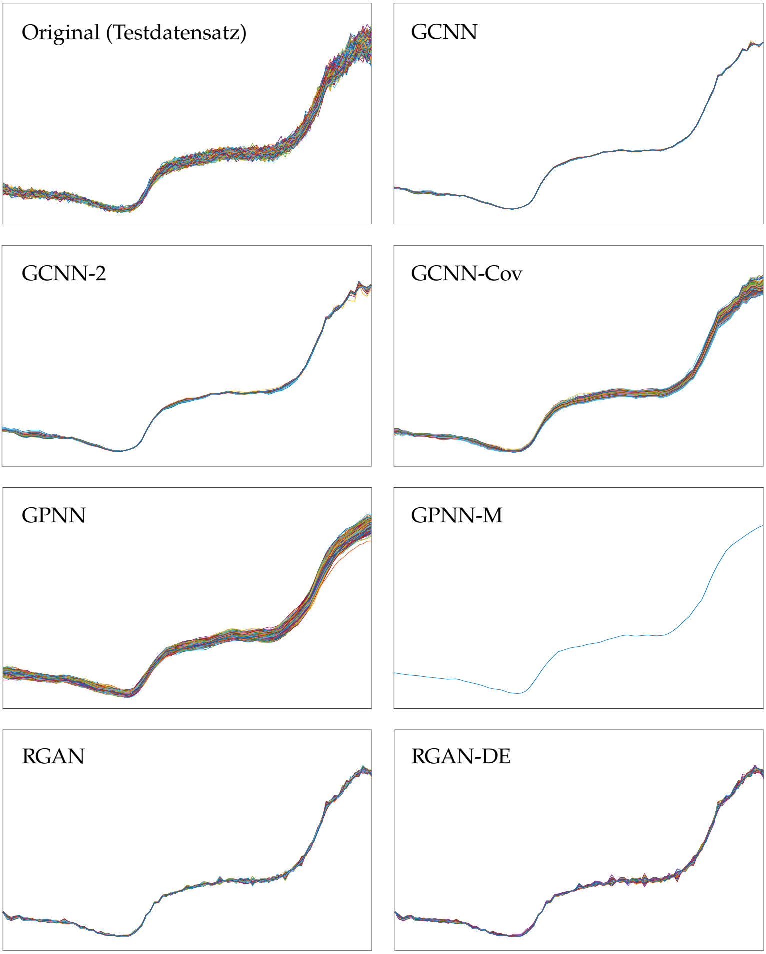


Abbildung B.10 Beispiele für mit unterschiedlichen Verfahren und dem Anteilsvektor $\mathbf{a} = [0{,}375\,;0{,}625\,;0\,]^{\mathrm{T}}$ generierte Spektren des Datensatzes $\mathcal{X}_{\mathrm{Q3}}$. Zu Gunsten der Lesbarkeit wird auf eine Achsenbeschriftung verzichtet. Die Achsen sind in allen Bildern gleich skaliert und die relative Reflektanz ist über der Wellenlänge von 450 nm bis 810 nm aufgetragen.

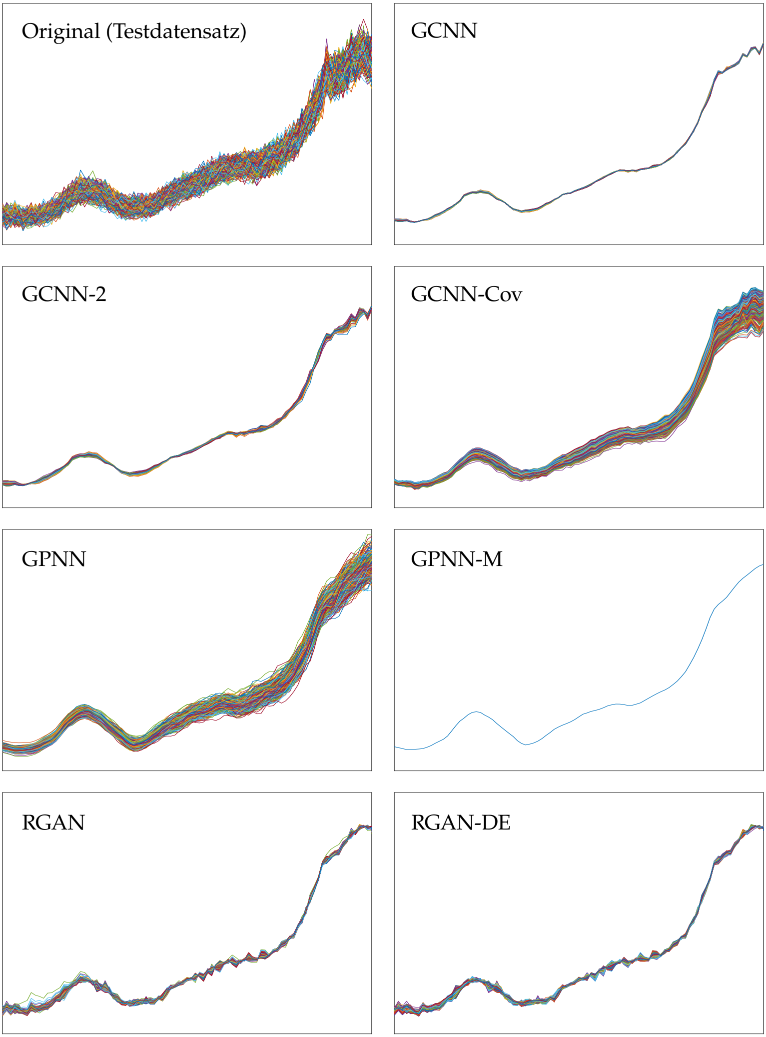


Abbildung B.11 Beispiele für mit unterschiedlichen Verfahren und dem Anteilsvektor $\mathbf{a} = [0{,}2\,;0{,}2\,;0{,}4\,;0{,}2\,]^T$ generierte Spektren des Datensatzes $\mathcal{X}_{Q4}$. Zu Gunsten der Lesbarkeit wird auf eine Achsenbeschriftung verzichtet. Die Achsen sind in allen Bildern gleich skaliert und die relative Reflektanz ist über der Wellenlänge von 450 nm bis 810 nm aufgetragen.

Literaturverzeichnis

[1] **D. Bank, N. Koenigstein und R. Giryes**. *Autoencoders*. In: *arXiv preprint arXiv:2003.05991* (2020).

[2] **I. F. Barton, M. J. Gabriel, J. Lyons-Baral, M. D. Barton, L. Duplessis und C. Roberts**. *Extending geometallurgy to the mine scale with hyperspectral imaging: a pilot study using drone- and ground-based scanning*. In: *Mining, Metallurgy & Exploration* 38 (2021), S. 799–818.

[3] **S. Bauer**. *Hyperspectral Image Unmixing Incorporating Adjacency Information*. Diss. Karlsruher Institut für Technologie (KIT), 2018.

[4] **S. Bauer, J. Stefan und F. Puente León**. *Hyperspectral image unmixing involving spatial information by extending the alternating least-squares algorithm*. In: *tm – Technisches Messen* 82.4 (2015), S. 174–186.

[5] **S. Bauer, W. Krippner, D. Luo, M. Taphanel, M. Abdo, V. Badilita, T. Längle, J. Beyerer, J. Korvink und F. Puente León**. *Ortsaufgelöste optische Bestimmung von Materialanteilen in Mischungen*. In: *XXX. Messtechnisches Symposium*. 2016, S. 69–76.

[6] **E. Ben-Dor, D. Schläpfer, A. J. Plaza und T. Malthus**. *Hyperspectral Remote Sensing*. In: *Airborne Measurements for Environmental Research: Methods and Instruments* (2013), S. 413–456.

[7] **R. A. Borsoi, T. Imbiriba und J. C. M. Bermudez**. *Deep Generative Endmember Modeling: An Application to Unsupervised Spectral Unmixing*. In: *IEEE Transactions on Computational Imaging* 6 (2020), S. 374–384.

[8] **R. A. Borsoi**. *Spectral Variability in Hyperspectral Unmixing: Multiscale, Tensor, and Neural Network-based Approaches*. Diss. Université Côte d'Azur; Universidade federal de Santa Catarina (Brésil), 2021.

[9] **R. A. Borsoi, T. Imbiriba, J. C. M. Bermudez und C. Richard**. *Deep Generative Models for Library Augmentation in Multiple Endmember Spectral Mixture Analysis*. In: *IEEE Geoscience and Remote Sensing Letters* 18.10 (2021), S. 1831–1835.

[10] **R. A. Borsoi, T. Imbiriba, J. C. M. Bermudez, C. Richard, J. Chanussot, L. Drumetz, J.-Y. Tourneret, A. Zare und C. Jutten**. *Spectral Variability in Hyperspectral Data Unmixing: A comprehensive review*. In: *IEEE Geoscience and Remote Sensing Magazine* 9.4 (2021), S. 223–270.

[11] **C. G. Broyden**. *Quasi-Newton Methods and their Application to Function Minimisation*. In: *Mathematics of Computation* 21.99 (1967), S. 368–381.

[12] **A. Cauchy**. *Méthode générale pour la résolution des systemes d'équations simultanées (Übersetzung von R. J. Pulskamp)*. In: *Comptes Rendus Hebd. Séances Acad. Sci.* 25 (1847), S. 536–538.

[13] **S. Chandrasekhar**. *Radiative Transfer*. Dover books on intermediate and advanced mathematics. New York: Dover Publications, 1960.

[14] **S. Chetlur, C. Woolley, P. Vandermersch, J. Cohen, J. Tran, B. Catanzaro und E. Shelhamer**. *cuDNN: Efficient Primitives for Deep Learning*. In: *arXiv preprint arXiv:1410.0759* (2014).

[15] **R. N. Clark und T. L. Roush**. *Reflectance Spectroscopy: Quantitative Analysis Techniques for Remote Sensing Applications*. In: *Journal of Geophysical Research* 89.B7 (1984), S. 6329–6340.

[16] **R. Close, P. Gader, J. Wilson und A. Zare**. *Using Physics-Based Macroscopic and Microscopic Mixture Models for Hyperspectral Pixel Unmixing*. In: *Algorithms and Technologies for Multispectral, Hyperspectral, and Ultraspectral Imagery XVIII*. 2012, 83901L.

[17] **E. D. Cubuk, B. Zoph, D. Mane, V. Vasudevan und Q. V. Le**. *AutoAugment: Learning Augmentation Policies from Data*. In: *arXiv preprint arXiv:1805.09501* (2018).

[18] **J. P. Dash, M. S. Watt, G. D. Pearse, M. Heaphy und H. S. Dungey**. *Assessing very high resolution UAV imagery for monitoring forest health during a simulated disease outbreak*. In: *ISPRS Journal of Photogrammetry and Remote Sensing* 131 (2017), S. 1–14.

[19] **L. Deng, G. Hinton und B. Kingsbury**. *New types of deep neural network learning for speech recognition and related applications: an overview*. In: *2013 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 2013, S. 8599–8603.

[20] **P. Dhariwal und A. Nichol**. *Diffusion Models Beat GANs on Image Synthesis*. In: *Advances in Neural Information Processing Systems 34*. 2021, S. 8780–8794.

[21] **X. Ding, Y. Wang, Z. Xu, W. J. Welch und Z. J. Wang**. *Continuous Conditional Generative Adversarial Networks: Novel Empirical Losses and Label Input Mechanisms*. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2022), S. 1–16.

[22] **N. Dobigeon, J.-Y. Tourneret, C. Richard, J. C. M. Bermudez, S. McLaughlin und A. O. Hero**. *Nonlinear Unmixing of Hyperspectral Images: Models and algorithms*. In: *IEEE Signal Processing Magazine* 31.1 (2014), S. 82–94.

[23] **N. Dobigeon, Y. Altmann, N. Brun und S. Moussaoui**. *Linear and Nonlinear Unmixing in Hyperspectral Imaging*. In: *Data Handling in Science and Technology*. Bd. 30. 2016, S. 185–224.

[24] **C. Doersch**. *Tutorial on Variational Autoencoders*. In: *arXiv preprint arXiv:1606.05908* (2016).

[25] **T. Dozat**. *Incorporating Nesterov Momentum Into Adam*. In: *International Conference on Learning Representations 2016 workshop submission*. 2016.

[26] **L. Drumetz, J. Chanussot und C. Jutten**. *Spectral Unmixing: A Derivation of the Extended Linear Mixing Model From the Hapke Model*. In: *IEEE Geoscience and Remote Sensing Letters* 17.11 (2020), S. 1866–1870.

[27] **L. Drumetz, M.-A. Veganzones, S. Henrot, R. Phlypo, J. Chanussot und C. Jutten**. *Blind Hyperspectral Unmixing Using an Extended Linear Mixing Model to Address Spectral Variability*. In: *IEEE Transactions on Image Processing* 25.8 (2016), S. 3890–3905.

[28] **J. Duchi, E. Hazan und Y. Singer**. *Adaptive Subgradient Methods for Online Learning and Stochastic Optimization.* In: *Journal of Machine Learning Research* 12 (2011), S. 2121–2159.

[29] **G. Elmasry, M. Kamruzzaman, D.-W. Sun und P. Allen**. *Principles and Applications of Hyperspectral Imaging in Quality Evaluation of Agro-Food Products: A Review*. In: *Critical Reviews in Food Science and Nutrition* 52.11 (2012), S. 999–1023.

[30] **W. Fan, B. Hu, J. Miller und M. Li**. *Comparative study between a new nonlinear model and common linear model for analysing laboratory simulated-forest hyperspectral data*. In: *International Journal of Remote Sensing* 30.11 (2009), S. 2951–2962.

[31] **Y. Fang, Y. Wang, L. Xu, R. Zhuo, A. Wong und D. A. Clausi**. *Bcun: Bayesian Fully Convolutional Neural Network for Hyperspectral Spectral Unmixing*. In: *IEEE Transactions on Geoscience and Remote Sensing* 60, 5523714 (2022).

[32] **H. Fischer**. *A History of the Central Limit Theorem: From Classical to Modern Probability Theory*. Springer, 2011.

[33] **P. Ghosh, S. K. Roy, B. Koirala, B. Rasti und P. Scheunders**. *Hyperspectral Unmixing Using Transformer Network*. In: *IEEE Transactions on Geoscience and Remote Sensing* 60, 5535116 (2022).

[34] **A. F. H. Goetz**. *Three decades of hyperspectral remote sensing of the Earth: A personal view*. In: *Remote sensing of environment* 113.S1 (2009), S. 5–16.

[35] **I. Goodfellow, Y. Bengio und A. Courville**. *Deep Learning*. http://www.deeplearningbook.org. MIT Press, 2016.

[36] **I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville und Y. Bengio**. *Generative Adversarial Nets*. In: *Advances in Neural Information Processing Systems* 27 (2014), S. 2672–2680.

[37] **A. A. Gowen, C. P. O'Donnell, P. J. Cullen, G. Downey und J. M. Frias**. *Hyperspectral imaging – an emerging process analytical tool for food quality and safety control*. In: *Trends in Food Science & Technology* 18.12 (2007), S. 590–598.

[38] **H. F. Grahn und P. Geladi**. *Techniques and Applications of Hyperspectral Image Analysis*. John Wiley & Sons, 2007.

[39] **M. K. Griffin und H.-h. K. Burke**. *Compensation of Hyperspectral Data for Atmospheric Effects*. In: *Lincoln Laboratory Journal* 14.1 (2003), S. 29–54.

[40] **R. Guo, W. Wang und H. Qi**. *Hyperspectral image unmixing using autoencoder cascade*. In: *7th Workshop on Hyperspectral Image and Signal Processing: Evolution in Remote Sensing (WHISPERS)*. 2015.

[41] **A. Halimi, Y. Altmann, N. Dobigeon und J.-Y. Tourneret**. *Nonlinear Unmixing of Hyperspectral Images Using a Generalized Bilinear Model*. In: *IEEE Transactions on Geoscience and Remote Sensing* 49.11 (2011), S. 4153–4162.

[42] **A. Halimi, Y. Altmann, N. Dobigeon und J.-Y. Tourneret**. *Unmixing hyperspectral images using the generalized bilinear model*. In: *2011 IEEE International Geoscience and Remote Sensing Symposium (IGARSS)*. 2011, S. 1886–1889.

[43] **B. Hapke**. *Theory of Reflectance and Emittance Spectroscopy*. 2. Aufl. Cambridge University Press, 2012.

[44] **T. Hastie, R. Tibshirani und J. H. Friedman**. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. 2. Aufl. Springer, 2009.

[45] **K. He, X. Zhang, S. Ren und J. Sun**. *Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification*. In: *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*. 2015, S. 1026–1034.

[46] **K. He, X. Zhang, S. Ren und J. Sun**. *Deep Residual Learning for Image Recognition*. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016, S. 770–778.

[47] **D. Heinz, C.-I. Chang und M. L. G. Althouse**. *Fully constrained least-squares based linear unmixing [hyperspectral image classification]*. In: *IEEE 1999 International Geoscience and Remote Sensing Symposium (IGARSS)*. Bd. 2. 1999, S. 1401–1403.

[48] **R. Heylen und P. Gader**. *Nonlinear Spectral Unmixing With a Linear Mixture of Intimate Mixtures Model*. In: *IEEE Geoscience and Remote Sensing Letters* 11.7 (2014), S. 1195–1199.

[49] **R. Heylen, M. Parente und P. Gader**. *A Review of Nonlinear Hyperspectral Unmixing Methods*. In: *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing* 7.6 (2014), S. 1844–1868.

[50] **G. Hinton, N. Srivastava und K. Swersky**. *Lecture 6d – A separate, adaptive learning rate for each connection.* In: *Neural Networks for Machine Learning* (2012).

[51] **J. Ho, A. Jain und P. Abbeel**. *Denoising Diffusion Probabilistic Models*. In: *Advances in Neural Information Processing Systems* 33 (2020), S. 6840–6851.

[52] **W. J. Holland und Q. Du**. *Adversarially regularized autoencoder for hyperspectral image unmixing*. In: *Image and Signal Processing for Remote Sensing XXVI*. 2020, 115330U.

[53] **Z. Hua, X. Li, Y. Feng und L. Zhao**. *Dual Branch Autoencoder Network for Spectral-Spatial Hyperspectral Unmixing*. In: *IEEE Geoscience and Remote Sensing Letters* 19, 5507305 (2022).

[54] **T. Imbiriba, R. A. Borsoi und J. C. Moreira Bermudez**. *Generalized Linear Mixing Model Accounting for Endmember Variability*. In: *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 2018, S. 1862–1866.

[55] **S. Ioffe und C. Szegedy**. *Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift*. In: *Proceedings of the 32nd International Conference on Machine Learning*. 2015, S. 448–456.

[56] **L. Isserlis**. *On a Formula for the Product-Moment Coefficient of any Order of a Normal Frequency Distribution in any Number of Variables*. In: *Biometrika* 12.1/2 (1918), S. 134–139.

[57] **S. Jay, F. Baret, D. Dutartre, G. Malatesta, S. Héno, A. Comar, M. Weiss und F. Maupas**. *Exploiting the centimeter resolution of UAV multispectral imagery to improve remote-sensing estimates of canopy structure and biochemistry in sugar beet crops*. In: *Remote Sensing of Environment* 231, 110898 (2019).

[58] **Q. Jin, Y. Ma, F. Fan, J. Huang, X. Mei und J. Ma**. *Adversarial Autoencoder Network for Hyperspectral Unmixing*. In: *IEEE Transactions on Neural Networks and Learning Systems* (2021), S. 1–15.

[59] **F. Jondral und A. Wiesler**. *Grundlagen der Wahrscheinlichkeitsrechnung und stochastischer Prozesse für Ingenieure*. Springer, 2000. Kap. 5.

[60] **N. Keshava und J. F. Mustard**. *Spectral unmixing*. In: *IEEE Signal Processing Magazine* 19.1 (2002), S. 44–57.

[61] **D. P. Kingma und J. Ba**. *Adam: A Method for Stochastic Optimization*. In: *arXiv preprint arXiv:1412.6980* (2014).

[62] **D. P. Kingma und M. Welling**. *Auto-Encoding Variational Bayes*. In: *arXiv preprint arXiv:1312.6114* (2013).

[63] **Y. Kodratoff und R. S. Michalsky**. *Machine Learning: An Artificial Intelligence Approach*. 3. Aufl. San Mateo: Kaufman Publishers Inc., 1990.

[64] **A. H. Kramer und A. Sangiovanni-Vincentelli**. *Efficient Parallel Learning Algorithms for Neural Networks*. In: *Advances in Neural Information Processing Systems* 1 (1988).

[65] **M. A. Kramer**. *Nonlinear Principal Component Analysis Using Autoassociative Neural Networks*. In: *AIChE Journal* 37.2 (1991), S. 233–243.

[66] **W. Krippner**. *Entwurfsverfahren für optische Spektralfilter zur Materialanteilsschätzung*. Diss. Karlsruher Institut für Technologie (KIT), 2021.

[67] **W. Krippner, S. Bauer und F. Puente León**. *Ortsaufgelöste optische Bestimmung von Materialanteilen in Mischungen*. In: *tm – Technisches Messen* 84.3 (2017), S. 207–215.

[68] **A. Krizhevsky, I. Sutskever und G. E. Hinton**. *ImageNet Classification with Deep Convolutional Neural Networks*. In: *Proceedings of the 25th International Conference on Neural Information Processing Systems*. 2012, S. 1097–1105.

[69] **S. Kullback und R. A. Leibler**. *On Information and Sufficiency*. In: *The Annals of Mathematical Statistics* 22.1 (1951), S. 79–86.

[70] **Y. LeCun, Y. Bengio und G. Hinton**. *Deep learning*. In: *nature* 521 (2015), S. 436–444.

[71] **Y. LeCun, K. Kavukcuoglu und C. Farabet**. *Convolutional Networks and Applications in Vision*. In: *Proceedings of 2010 IEEE International Symposium on Circuits and Systems*. 2010, S. 253–256.

[72] **Y. LeCun, L. Bottou, Y. Bengio und P. Haffner**. *Gradient-Based Learning Applied to Document Recognition*. In: *Proceedings of the IEEE* 86.11 (1998), S. 2278–2324.

[73] **C. Lemaréchal**. *Cauchy and the Gradient Method*. In: *Documenta Mathematica* Extra Volume ISMP (2012), S. 251–254.

[74] **G. Lu und B. Fei**. *Medical hyperspectral imaging: a review*. In: *Journal of Biomedical Optics* 19.1, 010901 (2014).

[75] **A. Makhzani, J. Shlens, N. Jaitly, I. Goodfellow und B. Frey**. *Adversarial Autoencoders*. In: *arXiv preprint arXiv:1511.05644* (2015).

[76] **J. Masci, U. Meier, D. Cireşan und J. Schmidhuber**. *Stacked Convolutional Auto-Encoders for Hierarchical Feature Extraction*. In: *International Conference on Artificial Neural Networks – ICANN 2011*. 2011, S. 52–59.

[77] **W. S. McCulloch und W. Pitts**. *A logical calculus of the ideas immanent in nervous activity*. In: *The Bulletin of Mathematical Biophysics* 5 (1943), S. 115–133.

[78] **F. D. van der Meer, H. M. A. van der Werff, F. J. A. van Ruitenbeek, C. A. Hecker, W. H. Bakker, M. F. Noomen, M. van der Meijde, E. J. M. Carranza, J. B. de Smeth und T. Woldai**. *Multi- and hyperspectral geologic remote sensing: A review*. In: *International Journal of Applied Earth Observation and Geoinformation* 14.1 (2012), S. 112–128.

[79] **I. Meganem, P. Déliot, X. Briottet, Y. Deville und S. Hosseini**. *Linear-Quadratic Mixing Model for Reflectances in Urban Environments*. In: *IEEE Transactions on Geoscience and Remote Sensing* 52.1 (2013), S. 544–558.

[80] **M. Mirza und S. Osindero**. *Conditional Generative Adversarial Nets*. In: *arXiv preprint arXiv:1411.1784* (2014).

[81] **N. Mitschke**. *Konvolutionäre neuronale Netze in der industriellen Bildverarbeitung und Robotik*. Diss. Karlsruher Institut für Technologie (KIT), 2021.

[82] **J. F. Mustard und C. M. Pieters**. *Quantitative Abundance Estimates From Bidirectional Reflectance Measurements*. In: *Journal of Geophysical Research: Solid Earth* 92.B4 (1987), E617–E626.

[83] **J. M. P. Nascimento und J. M. Bioucas-Dias**. *Nonlinear mixture model for hyperspectral unmixing*. In: *Image and Signal Processing for Remote Sensing XV*. 2009, 74770I.

[84] **J. M. P. Nascimento und J. M. Bioucas-Dias**. *Unmixing Hyperspectral Intimate Mixtures*. In: *Image and signal processing for remote sensing XVI*. 2010, 78300C.

[85] **A. B. Nassif, I. Shahin, I. Attili, M. Azzeh und K. Shaalan**. *Speech Recognition Using Deep Neural Networks: A Systematic Review*. In: *IEEE Access* 7 (2019), S. 19143–19165.

[86] **Y. E. Nesterov**. *A method for solving the convex programming problem with convergence rate* $O\left(\frac{1}{k^2}\right)$. In: 269.3 (1983), S. 543–547.

[87] **A. Q. Nichol und P. Dhariwal**. *Improved Denoising Diffusion Probabilistic Models*. In: *Proceedings of the 38th International Conference on Machine Learning*. 2021, S. 8162–8171.

[88] **R. A. Oliveira, R. Näsi, O. Niemeläinen, L. Nyholm, K. Alhonoja, J. Kaivosoja, L. Jauhiainen, N. Viljanen, S. Nezami, L. Markelin et al.** *Machine learning estimators for the quantity and quality of grass swards used for silage production using drone-based imaging spectrometry and photogrammetry*. In: *Remote Sensing of Environment* 246, 111830 (2020).

[89] **B. Palsson, J. R. Sveinsson und M. O. Ulfarsson**. *Blind Hyperspectral Unmixing Using Autoencoders: A Critical Comparison*. In: *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing* 15 (2022), S. 1340–1372.

[90] **B. Palsson, J. Sigurdsson, J. R. Sveinsson und M. O. Ulfarsson**. *Hyperspectral Unmixing Using a Neural Network Autoencoder*. In: *IEEE Access* 6 (2018), S. 25646–25656.

[91] **V. Papyan, Y. Romano und M. Elad**. *Convolutional Neural Networks Analyzed via Convolutional Sparse Coding*. In: *Journal of Machine Learning Research* 18 (2017), S. 1–52.

[92] **V. Papyan, J. Sulam und M. Elad**. *Working Locally Thinking Globally: Theoretical Guarantees for Convolutional Sparse Coding*. In: *IEEE Transactions on Signal Processing* 65.21 (2017), S. 5687–5701.

[93] **A. Pinkus**. *Approximation theory of the MLP model in neural networks*. In: *Acta Numerica* 8 (1999), S. 143–195.

[94] **J. Plaza, P. Martínez, R. Pérez und A. Plaza**. *Nonlinear Neural Network Mixture Models for Fractional Abundance Estimation in AVIRIS Hyperspectral Images*. In: *Proceedings of the 13th JPL Airborne Earth Science Workshop*. 2004, S. 201–212.

[95] **J. Plaza, A. Plaza, R. Pérez und P. Martínez**. *On the use of small training sets for neural network-based characterization of mixed pixels in remotely sensed hyperspectral images*. In: *Pattern Recognition* 42.11 (2009), S. 3032–3045.

[96] **B. T. Polyak**. *Some methods of speeding up the convergence of iteration methods*. In: *USSR Computational Mathematics and Mathematical Physics* 4.5 (1964), S. 1–17.

[97] **V. Popov, I. Vovk, V. Gogoryan, T. Sadekova und M. Kudinov**. *Grad-TTS: A Diffusion Probabilistic Model for Text-to-Speech*. In: *Proceedings of the 38th International Conference on Machine Learning*. 2021, S. 8599–8608.

[98] **F. Puente León**. *Messtechnik: Grundlagen, Methoden und Anwendungen*. 11. Aufl. Springer Vieweg, 2019. Kap. 4.

[99] **Y. Qian, F. Xiong, Q. Qian und J. Zhou**. *Spectral Mixture Model Inspired Network Architectures for Hyperspectral Unmixing*. In: *IEEE Transactions on Geoscience and Remote Sensing* 58.10 (2020), S. 7418–7434.

[100] **N. Raksuntorn und Q. Du**. *Nonlinear Spectral Mixture Analysis for Hyperspectral Imagery in an Unknown Environment*. In: *IEEE Geoscience and Remote Sensing Letters* 7.4 (2010), S. 836–840.

[101] **B. Rasti und B. Koirala**. *SUnCNN: Sparse Unmixing Using Unsupervised Convolutional Neural Network*. In: *IEEE Geoscience and Remote Sensing Letters* 19, 5508205 (2021).

[102] **B. Rasti, B. Koirala und P. Scheunders**. *HapkeCNN: Blind Nonlinear Unmixing for Intimate Mixtures Using Hapke Model and Convolutional Neural Network*. In: *IEEE Transactions on Geoscience and Remote Sensing* 60, 5536315 (2022).

[103] **B. Rasti, B. Koirala, P. Scheunders und P. Ghamisi**. *Spectral Unmixing Using Deep Convolutional Encoder-Decoder*. In: *2021 IEEE International Geoscience and Remote Sensing Symposium (IGARSS)*. 2021, S. 3829–3832.

[104] **B. Rasti, B. Koirala, P. Scheunders und P. Ghamisi**. *UnDIP: Hyperspectral Unmixing Using Deep Image Prior*. In: *IEEE Transactions on Geoscience and Remote Sensing* 60, 5504615 (2021).

[105] **W. Rawat und Z. Wang**. *Deep Convolutional Neural Networks for Image Classification: A Comprehensive Review*. In: *Neural computation* 29.9 (2017), S. 2352–2449.

[106] **G. Ristorto, F. Mazzetto, G. Guglieri und F. Quagliotti**. *Monitoring Performances and Cost Estimation of Multirotor Unmanned Aerial Systems in Precision Farming*. In: *2015 International Conference on Unmanned Aircraft Systems (ICUAS)*. 2015, S. 502–509.

[107] **Y. Roggo, A. Edmond, P. Chalus und M. Ulmschneider**. *Infrared hyperspectral imaging for qualitative analysis of pharmaceutical solid forms*. In: *Analytica Chimica Acta* 535.1-2 (2005), S. 79–87.

[108] **T. Roper und M. Andrews**. *Shadow modelling and correction techniques in hyperspectral imaging*. In: *Electronics Letters* 49.7 (2013), S. 458–460.

[109] **F. Rosenblatt**. *The perceptron: A probabilistic model for information storage and organization in the brain*. In: *Psychological Review* 65.6 (1958), S. 386–408.

[110] **D. E. Rumelhart, G. E. Hinton und R. J. Williams**. *Learning representations by back-propagating errors*. In: *nature* 323 (1986), S. 533–536.

[111] **A. L. Samuel**. *Some studies in machine learning using the game of checkers*. In: *IBM Journal of Research and Development* 3 (1959), S. 210–229.

[112] **D. Scherer, A. Müller und S. Behnke**. *Evaluation of Pooling Operations in Convolutional Architectures for Object Recognition*. In: *International Conference on Artificial Neural Networks – ICANN 2010*. 2010, S. 92–101.

[113] **H. Schulz-Mirbach**. *Constructing invariant features by averaging techniques*. In: *Proceedings of the 12th IAPR International Conference on Pattern Recognition, Conference C: Signal Processing*. Bd. 2. 1994, S. 387–390.

[114] **S. Shi, M. Zhao, L. Zhang und J. Chen**. *Variational Autoencoders for Hyperspectral Unmixing with Endmember Variability*. In: *2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 2021, S. 1875–1879.

[115] **S. Shi, L. Zhang, Y. Altmann und J. Chen**. *Deep Generative Model for Spatial–Spectral Unmixing With Multiple Endmember Priors*. In: *IEEE Transactions on Geoscience and Remote Sensing* 60, 5527214 (2022).

[116] **K. Simonyan und A. Zisserman**. *Very Deep Convolutional Networks for Large-Scale Image Recognition*. In: *arXiv preprint arXiv:1409.1556* (2014).

[117] **M. H. Skjelvareid, K. Heia, S. H. Olsen und S. K. Stormo**. *Detection of blood in fish muscle by constrained spectral unmixing of hyperspectral images*. In: *Journal of Food Engineering* 212 (2017), S. 252–261.

[118] **J. Sohl-Dickstein, E. Weiss, N. Maheswaranathan und S. Ganguli**. *Deep Unsupervised Learning using Nonequilibrium Thermodynamics*. In: *Proceedings of the 32nd International Conference on Machine Learning*. 2015, S. 2256–2265.

[119] **Y. Su, A. Marinoni, J. Li, J. Plaza und P. Gamba**. *Stacked Nonnegative Sparse Autoencoders for Robust Hyperspectral Unmixing*. In: *IEEE Geoscience and Remote Sensing Letters* 15.9 (2018), S. 1427–1431.

[120] **Y. Su, J. Li, A. Plaza, A. Marinoni, P. Gamba und S. Chakravortty**. *DAEN: Deep Autoencoder Networks for Hyperspectral Unmixing*. In: *IEEE Transactions on Geoscience and Remote Sensing* 57.7 (2019), S. 4309–4321.

[121] **I. Sutskever, J. Martens, G. Dahl und G. Hinton**. *On the importance of initialization and momentum in deep learning*. In: *Proceedings of the 30th International Conference on Machine Learning*. 2013, S. 1139–1147.

[122] **R. S. Sutton und A. G. Barto**. *Reinforcement Learning: An Introduction*. MIT Press, 1998.

[123] **Y. Taigman, M. Yang, M. Ranzato und L. Wolf**. *DeepFace: Closing the Gap to Human-Level Performance in Face Verification*. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2014, S. 1701–1708.

[124] **X. Tao, M. E. Paoletti, L. Han, Z. Wu, L. I. Jiménez, J. M. Haut, P. Ren, J. Plaza und A. Plaza**. *A New 3D Convolution Network for Hyperspectral Unmixing*. In: *2022 IEEE International Geoscience and Remote Sensing Symposium (IGARSS)*. 2022, S. 1620–1623.

[125] **X. Tao, M. E. Paoletti, L. Han, Z. Wu, P. Ren, J. Plaza, A. Plaza und J. M. Haut**. *A New Deep Convolutional Network for Effective Hyperspectral Unmixing*. In: *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing* 15 (2022), S. 6999–7012.

[126] **A. Tarantola und B. Valette**. *Generalized Nonlinear Inverse Problems Solved Using the Least Squares Criterion*. In: *Reviews of Geophysics and Space Physics* 20.2 (1982), S. 219–232.

[127] **L. F. Tatzel**. *Verbesserungen beim Laserschneiden mit Methoden des maschinellen Lernens*. Diss. Karlsruher Institut für Technologie (KIT), 2021.

[128] **P. Tatzer, M. Wolf und T. Panner**. *Industrial application for inline material sorting using hyperspectral imaging in the NIR range*. In: *Real-Time Imaging* 11.2 (2005), S. 99–107.

[129] **P. S. Thenkabail, J. G. Lyon und A. Huete**. *Hyperspectral Remote Sensing of Vegetation*. Boca Raton: CRC Press, 2012.

[130] **S. Theodoridis und K. Koutroumbas**. *Pattern Recognition*. 3. Aufl. Elsevier, 2006.

[131] **L. Tulczyjew, M. Kawulok, N. Longépé, B. Le Saux und J. Nalepa**. *A Multibranch Convolutional Neural Network for Hyperspectral Unmixing*. In: *IEEE Geoscience and Remote Sensing Letters* 19, 6011105 (2022).

[132] **M. A. Veganzones, L. Drumetz, G. Tochon, M. Dalla Mura, A. Plaza, J. Bioucas-Dias und J. Chanussot**. *A new extended linear mixing model to address spectral variability*. In: *2014 6th Workshop on Hyperspectral Image and Signal Processing: Evolution in Remote Sensing (WHISPERS)*. 2014.

[133] **S. S. Vijayashekhar, V. S. Deshpande und J. S. Bhatt**. *A Practical Approach for Hyperspectral Unmixing Using Deep Learning*. In: *IEEE Geoscience and Remote Sensing Letters* 19, 5511505 (2021).

[134] **L. Wan, T. Chen, A. Plaza und H. Cai**. *Hyperspectral Unmixing Based on Spectral and Sparse Deep Convolutional Neural Networks*. In: *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing* 14 (2021), S. 11669–11682.

[135] **M. Wang, M. Zhao, J. Chen und S. Rahardja**. *Nonlinear Unmixing of Hyperspectral Data via Deep Autoencoder Networks*. In: *IEEE Geoscience and Remote Sensing Letters* 16.9 (2019), S. 1467–1471.

[136] **W. W. Wendlandt und H. G. Hecht**. *Reflectance spectroscopy*. New York: Interscience Publ., 1966.

[137] **C. S. Withers**. *The moments of the multivariate normal*. In: *Bulletin of the Australian Mathematical Society* 32.1 (1985), S. 103–107.

[138] **F. Xiong, J. Zhou, S. Tao, J. Lu und Y. Qian**. *SNMF-Net: Learning a Deep Alternating Neural Network for Hyperspectral Unmixing*. In: *IEEE Transactions on Geoscience and Remote Sensing* 60, 5510816 (2021).

[139] **X. Xu, Z. Shi und B. Pan**. *A supervised abundance estimation method for hyperspectral unmixing*. In: *Remote Sensing Letters* 9.4 (2018), S. 383–392.

[140] **M. D. Zeiler**. *ADADELTA: An Adaptive Learning Rate Method*. In: *arXiv preprint arXiv:1212.5701* (2012).

[141] **G. Zhang, S. Mei, B. Xie, M. Ma, Y. Zhang, Y. Feng und Q. Du**. *Spectral Variability Augmented Sparse Unmixing of Hyperspectral Images*. In: *IEEE Transactions on Geoscience and Remote Sensing* 60, 5527413 (2022).

[142] **X. Zhang, Y. Sun, J. Zhang, P. Wu und L. Jiao**. *Hyperspectral Unmixing via Deep Convolutional Neural Networks*. In: *IEEE Geoscience and Remote Sensing Letters* 15.11 (2018), S. 1755–1759.

[143] **M. Zhao, L. Yan und J. Chen**. *LSTM-DNN Based Autoencoder Network for Nonlinear Hyperspectral Image Unmixing*. In: *IEEE Journal of Selected Topics in Signal Processing* 15.2 (2021), S. 295–309.

[144] **M. Zhao, M. Wang, J. Chen und S. Rahardja**. *Hyperspectral Unmixing for Additive Nonlinear Models With a 3-D-CNN Autoencoder Network*. In: *IEEE Transactions on Geoscience and Remote Sensing* 60, 5509415 (2021).

Eigene Veröffentlichungen

[V1] Pilar Hernández Mesa, **Johannes Anastasiadis** und Fernando Puente León. *Identification and Sorting of Regular Textures According to Their Similarity*. In: *Automated Visual Inspection and Machine Vision*. 2015, 95300A.

[V2] **Johannes Anastasiadis** und Fernando Puente León. *Detektion von Stoffen in Lebensmitteln mit Hilfe von 3D-Faltungsautoencodern*. In: *tm – Technisches Messen* 85.S1 (2018), S. 38–44.

[V3] **Johannes Anastasiadis**, Wolfgang Krippner und Fernando Puente Léon. *Spatially resolved ingredient detection in spice mixes using 3D convolutional neural networks*. In: *OCM 2019 – Optical Characterization of Materials: Conference Proceedings*. 2019, S. 35–44.

[V4] **Johannes Anastasiadis** und Fernando Puente León. *Ortsaufgelöste spektrale Entmischung mit Hilfe von konvolutionalen neuronalen Netzen*. In: *tm – Technisches Messen* 86.S1 (2019), S. 122–126.

[V5] Wolfgang Krippner, **Johannes Anastasiadis** und Fernando Puente León. *Robust iterative estimation of material abundances based on spectral filters exploiting the SVD*. In: *Algorithms, Technologies, and Applications for Multispectral and Hyperspectral Imagery XXV*. 2019, 109861T.

[V6] **Johannes Anastasiadis**, Philipp Benzing und Fernando Puente León. *Erzeugung künstlicher Datensätze zum Training konvolutionaler neuronaler Netze für die spektrale Entmischung*. In: *tm – Technisches Messen* 87.9 (2020), S. 542–552.

[V7] **Johannes Anastasiadis** und Michael Heizmann. *CNN-based augmentation strategy for spectral unmixing datasets considering spectral variability*. In: *Image and Signal Processing for Remote Sensing XXVI*. 2020, 115330V.

[V8] **Johannes Anastasiadis** und Michael Heizmann. *Gaussian Process Inspired Neural Networks for Spectral Unmixing Dataset Augmentation*. In: *Artificial Intellgence – Application in Life Sciences and Beyond. The Upper Rhine Artificial Intelligence Symposium UR-AI 2021*. 2021, S. 61–70.

[V9] **Johannes Anastasiadis** und Michael Heizmann. *Generation of artificial training data for spectral unmixing by modelling spectral variability using Gaussian random variables*. In: *OCM 2021 – Optical Characterization of Materials: Conference Proceedings*. 2021, S. 129–139.

[V10] **Johannes Anastasiadis** und Michael Heizmann. *GAN-regularized augmentation strategy for spectral datasets*. In: *tm – Technisches Messen* 89.4 (2022), S. 278–288.

[V11] Jannick Kuester, **Johannes Anastasiadis**, Wolfgang Middelmann und Michael Heizmann. *Investigating the influence of hyperspectral data compression on spectral unmixing*. In: *Image and Signal Processing for Remote Sensing XXVIII*. 2022, 122670H.

Betreute studentische Arbeiten

[S1] **S. Wang**. *Bearbeitung hyperspektraler Bilder mit Autoencodern*. Masterarbeit. 2018.

[S2] **M. Aizaz**. *Input learning for non-linear abundance estimation using a pre-trained neural network*. Masterarbeit. 2019.

[S3] **Y. Gao**. *Untersuchung hyperspektraler Fluoreszenzbilder von Saatgut*. Bachelorarbeit. 2019.

[S4] **Y. Pan**. *Hyperspectral Image Processing using Recurrent Neural Networks*. Masterarbeit. 2019.

[S5] **T. Schneble**. *Detektion von fremden Substanzen in Stoffgemischen mit neuronalen Netzen*. Masterarbeit. 2019.

[S6] **X. Su**. *Vorverarbeitung hyperspektraler Bilder für die spektrale Entmischung mit neuronalen Netzen*. Bachelorarbeit. 2019.

[S7] **Y. Yao**. *Spectral unmixing using recurrent neural networks*. Bachelorarbeit. 2019.

[S8] **S. Braun**. *Discovering Part Segmentations Without Supervision or Constraints*. Masterarbeit. 2020.

[S9] **J. Jazewitsch**. *Vearbeitung hyperspektraler Bilder mit variational Autoencodern*. Masterarbeit. 2020.

[S10] **Z. Qiang**. *Processing of hyperspectral images using Kervolutional Neural Networks*. Masterarbeit. 2020.

[S11] **J. Chen**. *Defense against Adversarial Inputs in Deep Learning Models*. Masterarbeit. 2021.

[S12] **L. Freyler**. *Reinspektrenextraktion aus Hyperspektralbildern mithilfe von Autoencodern*. Bachelorarbeit. 2021.

Forschungsberichte aus der Industriellen Informationstechnik (ISSN 2190-6629)

Band 1 Pérez Grassi, Ana
Variable illumination and invariant features for detecting and classifying varnish defects.
ISBN 978-3-86644-537-6

Band 2 Christ, Konrad
Kalibrierung von Magnet-Injektoren für Benzin-Direkteinspritzsysteme mittels Körperschall.
ISBN 978-3-86644-718-9

Band 3 Sandmair, Andreas
Konzepte zur Trennung von Sprachsignalen in unterbestimmten Szenarien.
ISBN 978-3-86644-744-8

Band 4 Bauer, Michael
Vergleich von Mehrträger-Übertragungsverfahren und Entwurfskriterien für neuartige Powerline-Kommunikationssysteme zur Realisierung von Smart Grids.
ISBN 978-3-86644-779-0

Band 5 Kruse, Marco
Mehrobjekt-Zustandsschätzung mit verteilten Sensorträgern am Beispiel der Umfeldwahrnehmung im Straßenverkehr.
ISBN 978-3-86644-982-4

Band 6 Dudeck, Sven
Kamerabasierte In-situ-Überwachung gepulster Laserschweißprozesse.
ISBN 978-3-7315-0019-3

Band 7 Liu, Wenqing
Emulation of Narrowband Powerline Data Transmission Channels and Evaluation of PLC Systems.
ISBN 978-3-7315-0071-1

Band 8 Otto, Carola
Fusion of Data from Heterogeneous Sensors with Distributed Fields of View and Situation Evaluation for Advanced Driver Assistance Systems.
ISBN 978-3-7315-0073-5

Band 9 Wang, Limeng
Image Analysis and Evaluation of Cylinder Bore Surfaces in Micrographs.
ISBN 978-3-7315-0239-5

Band 10 Michelsburg, Matthias
Materialklassifikation in optischen Inspektionssystemen mithilfe hyperspektraler Daten.
ISBN 978-3-7315-0273-9

Band 11 Pallauf, Johannes
Objektsensitive Verfolgung und Klassifikation von Fußgängern mit verteilten Multi-Sensor-Trägern.
ISBN 978-3-7315-0529-7

Band 12 Sigle, Martin
Robuste Schmalband-Powerline-Kommunikation für Niederspannungsverteilernetze.
ISBN 978-3-7315-0539-6

Band 13 Opalko, Oliver
Powerline-Kommunikation für Batteriemanagement-Systeme in Elektro- und Hybridfahrzeugen.
ISBN 978-3-7315-0647-8

Band 14 Han, Bin
Characterization and Emulation of Low-Voltage Power Line Channels for Narrowband and Broadband Communication.
ISBN 978-3-7315-0654-6

Band 15 Alonso, Damián Ezequiel
Wireless Data Transmission for the Battery Management System of Electric and Hybrid Vehicles.
ISBN 978-3-7315-0670-6

Band 16 Hernández Mesa, Pilar
Design and analysis of a content-based image retrieval system.
ISBN 978-3-7315-0692-8

Band 17 Suchaneck, André
Energiemanagement-Strategien für batterieelektrische Fahrzeuge.
ISBN 978-3-7315-0773-4

Band 18 Bauer, Sebastian
Hyperspectral Image Unmixing Incorporating Adjacency Information.
ISBN 978-3-7315-0788-8

Band 19 Vater, Sebastian
Monokulare Blickrichtungsschätzung zur berührungslosen Mensch-Maschine-Interaktion.
ISBN 978-3-7315-0789-5

Band 20 Back, Kristine
Erkennung menschlicher Aktivitäten durch Erfassung und Analyse von Bewegungstrajektorien.
ISBN 978-3-7315-0909-7

Band 21 Nürnberg, Thomas
Entwurf von Computational-Imaging-Systemen am Beispiel der monokularen Tiefenschätzung.
ISBN 978-3-7315-0941-7

Band 22 Kaiser, Cornelius
Adaptive Modulationsverfahren für die schmalbandige Powerline-Kommunikation in Niederspannungsnetzen.
ISBN 978-3-7315-1010-9

Band 23 Struckmeier, Frederick
Prozesssicherheit von Laserschneidmaschinen – Auflagemessung und Schachtelung
ISBN 978-3-7315-1127-4

Band 24 Tatzel, Leonie Felica
Verbesserungen beim Laserschneiden mit Methoden des maschinellen Lernens
ISBN 978-3-7315-1128-1

Die Bände sind unter www.ksp.kit.edu als PDF frei verfügbar oder als Druckausgabe bestellbar.

Band 25 Wetzel, Johannes
Probabilistic Models and Inference for Multi-View People Detection in Overlapping Depth Images
ISBN 978-3-7315-1177-9

Band 26 Mitschke, Norbert
Konvolutionäre neuronale Netze in der industriellen Bildverarbeitung und Robotik
ISBN 978-3-7315-1197-7

Band 27 Schambach, Maximilian
Reconstruction from Spatio-Spectrally Coded Multispectral Light Fields
ISBN 978-3-7315-1210-3

Band 28 Bächle, Matthias
Model-based Filtering of Interfering Signals in Ultrasonic Time Delay Estimations
ISBN 978-3-7315-1252-3

Band 29 Ismet Demirdelen
Nutzung der keramischen Glühkerze als Sensorelement
ISBN 978-3-7315-1265-3

Band 30 Johannes Anastasiadis
Überwachte Methoden für die spektrale Entmischung mit künstlichen neuronalen Netzen
ISBN 978-3-7315-1305-6

Karlsruher Institut für Technologie

ISSN 2190-6629
ISBN 978-3-7315-1305-6

Gedruckt auf FSC-zertifiziertem Papier